

[illegible]

DQC  
VO4

```

LL               IIIIII             SSSSSSSS
LL              IIIIIII            SSSSSSSS
LL              II                 SS
LL              II                 SS
LL              II                 SS
LL              II                 SS
LL              II                 SSSSSS
LL              II                 SSSSSS
LL              II                 SS
LL              II                 SS
LL              II                 SS
LL              II                 SS
LLLLLLLLLLLLLLL IIIIIII           SSSSSSSS
LLLLLLLLLLLLLLL IIIIIII           SSSSSSSS

```



(1)	73	PROGRAM ABSTRACT
(1)	122	EXTERNAL DEFINITIONS
(1)	147	LOCAL MACRO DEFINITIONS
(1)	239	LOCAL SYMBOLS AND UCB EXTENSIONS
(1)	406	STANDARD TABLES
(1)	511	FUNCTION DECISION TABLES
(1)	625	START I/O ROUTINE
(1)	816	RETRIABLE ERROR ANALYSIS
(1)	912	FATAL ERROR ANALYSIS
(1)	966	FUNCTION COMPLETION
(1)	997	HARDWARE FUNCTION DISPATCH
(1)	1071	IMEDIATE FUNCTION EXECUTION
(1)	1124	RECALIBRATE FUNCTION EXECUTION
(1)	1167	POSITIONING FUNCTION EXECUTION
(1)	1240	TRANSFER FUNCTION EXECUTION
(1)	1370	TRANSFER POST PROCESSING
(1)	1404	DATA CHECK AND PARAMETER UPDATE
(1)	1472	SPECIAL CONDITION (POWER, TIMEOUT)
(1)	1500	HARDWARE FUNCTION EXIT PROCESSING
(1)	1594	INTERRUPT SERVICE ROUTINE
(1)	1702	REGISTER SAVE ROUTINE
(1)	1781	UNEXPECTED INTERRUPT HANDLER
(1)	1829	GET STATUS, RESET, READ HEADER
(1)	1929	WAIT FOR CONTROLLER READY
(1)	1963	UNIT INITIALIZATION ROUTINE
(1)	2062	DRIVE CLASSIFICATION ROUTINE
(1)	2119	CONTROLLER INITIALIZATION ROUTINE
(1)	2157	UNIT DELIVERY ROUTINE
(1)	2199	REGISTER DUMP ROUTINE

```
0000 1 .TITLE DQDRIVER - VAX/VMS RB730:RB02/RB80 DISK DRIVER
0000 2 .IDENT 'V04-000'
0000 3
0000 4 *****
0000 5
0000 6 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 * ALL RIGHTS RESERVED.
0000 9
0000 10 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 * TRANSFERRED.
0000 16
0000 17 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 * CORPORATION.
0000 20
0000 21 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23
0000 24 *****
0000 25
0000 26
0000 27
0000 28 FACILITY:
0000 29
0000 30 VAX/VMS RB730:RB02/RB80 DISK DRIVER
0000 31
0000 32 AUTHOR:
0000 33
0000 34 G. ROBERT 21-JAN-1981
0000 35
0000 36 MODIFIED BY:
0000 37
0000 38 V03-008 RAS0300 Ron Schaefer 19-Jun-1984
0000 39 Add DEV$M_NNM characteristic to DECHAR2 so that these
0000 40 devices will have the "node$" prefix.
0000 41
0000 42 V03-007 ROW0211 Ralph O. Weber 28-DEC-1983
0000 43 Change device-dependent UCB definition base from UCBSW_BCR+2
0000 44 to UCBSK_LCL_DISK_LENGTH.
0000 45
0000 46 V03-006 PRD0035 Paul R. DeStefano 09-Sep-1983
0000 47 Added EXE$LCLDSKVALID to function decision table.
0000 48
0000 49 V03-005 PRD0026 Paul R. DeStefano 28-Jul-1983
0000 50 Modified ECC correction logic so that ECC is only applied
0000 51 when there is a single bit ECC correctable error, or if there
0000 52 is a multiple bit ECC correctable error and the error cannot
0000 53 be corrected using retries.
0000 54
0000 55 V03-004 PRD0025 Paul R. DeStefano 22-Jun-1983
0000 56 Modified FATALERR routine to return $$$_PARITY only for
0000 57 errors that possibly indicate bad media. All other error
```



0000 58 :  
0000 59 :  
0000 60 :  
0000 61 :  
0000 62 :  
0000 63 :  
0000 64 :  
0000 65 :  
0000 66 :  
0000 67 :  
0000 68 :  
0000 69 :  
0000 70 :  
0000 71 : \*\*

conditions which formerly returned SSS\_PARITY now return  
SSS\_CNTLERR.

V03-003 GRR3003 GREG ROBERT 16-SEP-1982  
RECORD PREVIOUS DISK ADDRESS IN ERROR LOG BUFFERS.  
V03-002 KDM0002 Kathleen D. Morse 28-Jun-1982  
Added \$DYNDEF.  
V03-001 KTA0100 Kerbey T. Altmann 07-Jun-1982  
Add code to set UCB\$MEDIA\_ID.

```
0000 73 .SBTTL PROGRAM ABSTRACT
0000 74
0000 75 : ABSTRACT:
0000 76 :
0000 77 : THIS MODULE CONTAINS THE TABLES AND ROUTINES NECESSARY TO
0000 78 : PERFORM ALL DEVICE-DEPENDENT PROCESSING OF AN I/O REQUEST
0000 79 : FOR RB730:RB02/RB80 DISK TYPES ON A VAX/VMS SYSTEM.
0000 80 :
0000 81 : THE DISKS HAVE THE FOLLOWING PHYSICAL GEOMETRY:
0000 82 :
0000 83 :
0000 84 : # CYL          TRACKS/      SECTORS/      BYTES/      MAXIMUM
0000 85 :                   CYLINDER    TRACK        SECTOR      BLOCKS
0000 86 : RB02          512             2           40         256      20480
0000 87 : RB80          561            14          32         512     251328
0000 88 :
0000 89 : SINCE THE RB02 SECTOR SIZE IS ONLY 1/2 BLOCK, LOGICAL TO PHYSICAL
0000 90 : CONVERSION OF RB02 DISK ADDRESSES BY IOC$CVTLOGPHY IS DELAYED
0000 91 : UNTIL STARTIO IS CALLED, AND THE DISK ADDRESS IS DOUBLED PRIOR
0000 92 : TO CONVERSION.
0000 93 :
0000 94 : ON THE RB80, THE LAST SECTOR IN EVERY TRACK IS RESERVED FOR
0000 95 : "SKIP SECTORING", AND THE LAST TWO CYLINDERS ARE RESERVED
0000 96 : FOR FIELD SERVICE. THE USER AVAILABLE RB80 GEOMETRY IS THEREFORE:
0000 97 :
0000 98 : RB80          559             14           31         512     242606
0000 99 :
0000 100 :
0000 101 : THE CONTROLLER DOES NOT READ OR WRITE BEYOND THE END OF TRACK
0000 102 : (SPIRALLING), SO READ AND WRITE FUNCTIONS ARE BROKEN UP BY THIS
0000 103 : DRIVER INTO PARTIAL TRANSFERS TO THE END OF TRACK, FOLLOWED BY
0000 104 : AN EXPLICIT SEEK TO THE NEXT TRACK, THEN ANOTHER READ OR
0000 105 : WRITE FUNCTION UNTIL THE TOTAL DATA TRANSFER IS COMPLETE.
0000 106 : (TRACK TO TRACK SPIRALLING FOR R80'S, WITHIN A CYLINDER,
0000 107 : IS DONE INSIDE THE XFER ROUTINE BY WRITING THE DAR).
0000 108 :
0000 109 : THE IOS_INHSEEK MODIFIER IS IGNORED BY THIS DRIVER.
0000 110 :
0000 111 : THE R02 DRIVE ON AN RB730 CONTROLLER IS CALLED AN RB02. THE
0000 112 : SAME DRIVE ON AN RL11 CONTROLLER IS KNOWN AS AN RL02. SIMILARLY
0000 113 : THE R80 DRIVE IS KNOWN AS THE RM80, RA80, AND RB80 WHEN PLACED
0000 114 : ON DIFFERENT CONTROLLERS. DRIVE DEPENDENT CHARACTERISTICS (SPEED,
0000 115 : SIZE, MECHANICAL TIMINGS) REMAIN THE SAME. CONTROLLER DEPENDENT
0000 116 : CHARACTERISTICS (COMMANDS, COMMAND TIMINGS, ERROR REPORTING) VARY
0000 117 : FROM CONTROLLER TO CONTROLLER.
0000 118 :
0000 119 :
0000 120 :--
```



```

0000 122      .SBTTL  EXTERNAL DEFINITIONS
0000 123
0000 124      :
0000 125      : EXTERNAL SYMBOLS
0000 126      :
0000 127
0000 128      $ADPDEF      ;DEFINE ADAPTER CONTROL BLOCK
0000 129      $CRBDEF      ;DEFINE CHANNEL REQUEST BLOCK
0000 130      $DCDEF       ;DEFINE DEVICE CLASS
0000 131      $DDBDEF      ;DEFINE DEVICE DATA BLOCK
0000 132      $DEVDEF      ;DEFINE DEVICE CHARACTERISTICS
0000 133      $DPTDEF      ;DEFINE DRIVER PROLOGUE TABLE
0000 134      $DYNDEF      ;DEFINE DYNAMIC DATA STRUCTURE TYPES
0000 135      $EMBDEF      ;DEFINE ERROR MESSAGE BUFFER
0000 136      $IDBDEF      ;DEFINE INTERRUPT DATA BLOCK
0000 137      $IODEF       ;DEFINE I/O FUNCTION CODES
0000 138      $IPLDEF      ;DEFINE IPL CODES
0000 139      $IRPDEF      ;DEFINE I/O REQUEST PACKET
0000 140      $PRDEF       ;DEFINE PROCESSOR REGISTERS
0000 141      $SSDEF       ;DEFINE SYSTEM STATUS CODES
0000 142      $SUBDEF      ;DEFINE UNIBUS ADAPTOR OFFSETS
0000 143      $UCBDEF      ;DEFINE UNIT CONTROL BLOCK
0000 144      $VECDEF      ;DEFINE INTERRUPT VECTOR BLOCK
0000 145

```

```
0000 147      .SBTTL  LOCAL MACRO DEFINITIONS
0000 148
0000 149      :
0000 150      : LOCAL MACROS
0000 151      :
0000 152      .MACRO  REQDPNRW
0000 153      JSB      G^IOCSREQDATAPNW
0000 154      .ENDM   REQDPNRW
0000 155
0000 156      .MACRO  LOADUBAA
0000 157      JSB      G^IOCSLOADUBAMAPA
0000 158      .ENDM   LOADUBAA
0000 159
0000 160      :
0000 161      : EXFUNCL
0000 162      : BRANCH TO SUBROUTINE WHICH REQUESTS CHANNEL (IF NOT ALREADY OWNED),
0000 163      : EXECUTES FCODE (OR R3) FUNCTION, AND BRANCHES TO BDST ON ERROR
0000 164      :
0000 165
0000 166      .MACRO  EXFUNCL  BDST,FCODE
0000 167      .IF  NB  FCODE      :IS FCODE NON-BLANK?
0000 168      MOVZBL  #CD'FCODE,R3  :IF NB - SPECIFY FCODE FUNCTION
0000 169      .ENDC      :IF B - SPECIFY FNTN IN EXISTING R3
0000 170      BSBW    FEXL      :EXECUTE FUNCTION
0000 171      .BYTE    BDST-.-1  :WHERE TO GO IF ERROR
0000 172      .ENDM
0000 173
0000 174      :
0000 175      : GENF
0000 176      : GENERATE FUNCTION TABLE ENTRY AND CASE TABLE INDEX SYMBOL
0000 177      :
0000 178      .MACRO  GENF  FCODE,MODS
0000 179      CD'FCODE=-FTAB/4
0000 180      _TMP$VAL = FCODE
0000 181      _TMP$VAL = _TMP$VAL ! RB_CS_M_IE
0000 182      .IRP  MODBIT,MODS
0000 183      _TMP$VAL = _TMP$VAL ! RB_CS_M_'MODBIT'
0000 184      .ENDR
0000 185      .LONG    _TMP$VAL
0000 186      .ENDM
0000 187
0000 188
0000 189      :
0000 190      : CKPWR
0000 191      : DISABLE INTERRUPTS, CHECK IF POWER HAS FAILED,
0000 192      :
0000 193      .MACRO  CKPWR,DEST=RETREG,?L1
0000 194
0000 195      SETIPL  #IPL$ POWER      :RAISE TO POWER
0000 196      BBC      #UCBS$ POWER,- :IF CLR - NO POWER FAILURE
0000 197      UCBSW_ STS(R5),L1      :
0000 198      ENBINT      :POWER FAILURE - RETURN TO SAVED IPL
0000 199      BRB      DEST      :EXIT
0000 200      L1:      :RETURN FOR NO POWER FAILURE
0000 201      .ENDM
0000 202
0000 203
```



```
0000 204 :  
0000 205 :GETUNIT  
0000 206 : GET UNIT NUMBER FROM UCB, PLACE IN SPECIFIED LOCATION OR  
0000 207 : R2 BY DEFAULT  
0000 208 :  
0000 209 :.MACRO GETUNIT,DEST=R2  
0000 210 :CLRL DEST ;CLEAR DEST FOR UNIT NUMBER  
0000 211 :INSV UCBSW_UNIT(R5), #8,#2,DEST ;PUT UNIT NUMBER IN DEST  
0000 212 :.ENDM  
0000 213 :  
0000 214 :  
0000 215 :  
0000 216 :INITIATE  
0000 217 : INITIATE A HARDWARE FUNCTION BY CLEARING CONTROLLER READY.  
0000 218 : PRESERVE THE ATTENTION AND INTERRUPT PENDING BITS BY CLEARING  
0000 219 : THEM (SINCE THEY ARE 'WRITE ONES TO CLEAR' THE FOLLOWING  
0000 220 : INSTRUCTION LEAVES THEM UNMODIFIED).  
0000 221 :  
0000 222 :.MACRO INITIATE  
0000 223 :BICL #RB_CS_M_CRDY- ;CLEAR CONTROLLER READY  
0000 224 :!RB_CS_M_ATN- ;...AND PRESERVE ATTENTION BITS  
0000 225 :!RB_CS_M_IR,- ;...AND INTERRUPT REQUEST BIT  
0000 226 :RB_CS(R4) ;...IN THE CSR  
0000 227 :.ENDM  
0000 228 :  
0000 229 :  
0000 230 :BDRVTYP  
0000 231 : BRANCH ON DRIVE TYPE  
0000 232 :  
0000 233 :.MACRO BDRVTYP TYPE,DEST  
0000 234 :CMPB #DTS_'TYPE,UCBSB_DEVTYPE(R5) ;COMPARE DRIVE TYPE  
0000 235 :BEQL DEST ;BRANCH IF SPECIFIED TYPE  
0000 236 :.ENDM  
0000 237 :
```

```
0000 239 .SBTTL LOCAL SYMBOLS AND UCB EXTENSIONS
0000 240
0000 241 :
0000 242 : LOCAL SYMBOLS
0000 243 :
00000007 0000 244 RB_NUM_REGS =7 ;NUMBER OF DEVICE REGISTERS
0000 245 ;(DOES NOT INCLUDE COMMAND REG (REG 8))
00000005 0000 246 RB_MP_C_SLM =5 ;STATE=SEEK LINEAR MODE (READY TO GO)
0000 247
0000 248 :
0000 249 : UCB OFFSETS WHICH FOLLOW THE STANDARD UCB FIELDS
0000 250 :
0000 251 $DEFINI UCB ;START OF UCB DEFINITIONS
000000C9 0000 252 .=UCB$W_OFFSET+1 ;REDEFINE FOR LOCAL USE
00C9 253
00C9 254 $DEF UCB$B_DQ_FLAGS ;LOCAL DRIVER FLAGS
00C9 255 $VIELD UCB,0,<- ;START OF DQ FLAGS DEFINITIONS
00C9 256 <DQ_SIP,,M>,- ;SEEK IN PROGRESS
00C9 257 <DQ_DIP,,M>,- ;DATA CHECK IN PROGRESS
00C9 258 <DQ_ECC_DEFER,,M>,- ;ECC CORRECTION DEFERRED TILL AFTER
00C9 259 > ;RETRY ATTEMPT
00C9 260 ;END OF DQ FLAGS BIT DEFINITIONS
000000CC 00C9 261
00C9 262 .=UCB$K_LCL_DISK_LENGTH
00CC 263
00CC 264 :
00CC 265 :ADJACENCY OF UCB EXTENSIONS ASSUMED BY DQ_REGDUMP AND READ HEADER CODE
00CC 266 :
00CC 267
00CC 268 $DEF UCB$S_DQ_CS .BLKL 1 ;CONTROL STATUS REGISTER
00D0 269 $DEF UCB$S_DQ_BA .BLKL 1 ;BUS ADDRESS REGISTER
00D4 270 $DEF UCB$S_DQ_BC .BLKL 1 ;BYTE COUNT REGISTER
00D8 271 $DEF UCB$S_DQ_DA .BLKL 1 ;DISK ADDRESS REGISTER
00DC 272 $DEF UCB$S_DQ_MP .BLKL 1 ;MULTIPURPOSE REGISTER
00E0 273 $DEF UCB$S_DQ_FMPR .BLKL 1 ;FINAL MAP REGISTER
00E4 274 $DEF UCB$S_DQ_PMPR .BLKL 1 ;PREVIOUS MAP REGISTER
00E8 275 $DEF UCB$S_DQ_DPR .BLKL 1 ;DATAPATH REGISTER (NEVER LOADED)
00EC 276 $DEF UCB$W_DQ_HDR1 .BLKW 1 ;SAVED HEADER WORD 1
00EE 277 $DEF UCB$W_DQ_HDR2 .BLKW 1 ;SAVED HEADER WORD 2
00F0 278 $DEF UCB$W_DQ_HDR3 .BLKW 1 ;SAVED HEADER WORD 3
00F2 279 $DEF UCB$S_DQ_CURDA .BLKL 1 ;CURRENT DISK ADDRESS
00F6 280 $DEF UCB$S_DQ_PREVDA .BLKL 1 ;PREVIOUS DISK ADDRESS
00FA 281
000000FA 00FA 282 UCB$K_DQ_LEN = . ;LENGTH OF EXTENDED UCB
00FA 283 $DEFEND UCB ;END OF UCB DEFINITIONS
0000 284
0000 285 :
0000 286 : RB730:RB02/RB80 REGISTER OFFSETS FROM CSR ADDRESS
0000 287 :
0000 288 $DEFINI RB ; START OF REGISTER DEFINITIONS
0000 289
0000 290 $DEF RB_CS .BLKL 1 ;CONTROL STATUS REGISTER (CSR)
0004 291 $VIELD RB_CS,0,<- ;START OF CSR BIT DEFINITIONS
0004 292 <DRDY,,M>,- ;DRIVE READY
0004 293 <FCODE,3,M>,- ;FUNCTION CODE
0004 294 <2>,- ;RESERVED BITS
0004 295 <IE,,M>,- ; INTERRUPT ENABLE
```



```
0004 296 <CRDY,,M>,- : CONTROLLER READY
0004 297 <DS,2,M>,- : DRIVE SELECT
0004 298 <OP1,,M>,- : OPERATION INCOMPLETE
0004 299 <DCK,,M>,- : DATA CRC OR HEADER CRC OR DATA ECC
0004 300 <DLT,,M>,- : DATA LATE OR HEADER NOT FOUND
0004 301 <NXM,,M>,- : NON-EXISTENT MEMORY
0004 302 <DE,,M>,- : DRIVE ERROR
0004 303 <CE,,M>,- : COMPOSITE ERROR
0004 304 <ATN,4,M>,- : DRIVE ATTENTION BITS
0004 305 <ECS,2>,- : ECC STATUS
0004 306 <SSE1,,M>,- : SKIP SECTOR ERROR INHIBIT
0004 307 <SSE,,M>,- : SKIP SECTOR ERROR
0004 308 <IR,,M>,- : RB730 INTERRUPT REQUEST
0004 309 <MTN,,M>,- : MAINTENANCE MODE
0004 310 <TYP,,M>,- : DRIVE TYPE 1=RB80, 0=RB02
0004 311 <ASS1,,M>,- : AUTOMATIC SKIP SECTOR INHIBIT
0004 312 <TO1,,M>,- : TIME OUT INHIBIT (U-DIAG'S)
0004 313 <FMT,,M>,- : R80 FORMAT CONTROL
0004 314 <,2>- : RESERVED BITS
0004 315 > : END CSR BIT DEFINITIONS
0004 316
0004 317 $DEF RB_BA .BLKL 1 :BUS ADDRESS REGISTER (BAR)
0008 318
0008 319 $DEF RB_BC .BLKL 1 :BYTE COUNT REGISTER (BCR)
000C 320
000C 321 $DEF RB_DA .BLKL 1 :DISK ADDRESS REGISTER (DAR)
0010 322 _VIELD RB_DA,0,<- :START OF DAR BIT DEFINITIONS
0010 323 <SEC,8>,- : SECTOR
0010 324 <TRK,8>,- : TRACK
0010 325 <CYL,16>- : CYLINDER
0010 326 > :END OF DAR BIT DEFINITIONS
0010 327
0010 328 $DEF RB_MP .BLKL 1 :MULTIPURPOSE REGISTER (MPR)
0014 329 _VIELD RB_MP,0,<- :RB02 STATUS WORD DEFINITIONS
0014 330 <STA,3>,- : DRIVE STATE
0014 331 <BH,,M>,- : BRUSH HOME
0014 332 <HO,,M>,- : HEADS OUT
0014 333 <CO,,M>,- : COVER OPEN
0014 334 <HS,,M>,- : HEAD SELECT
0014 335 <,1>- : RESERVED
0014 336 <DSE,,M>,- : DRIVE SELECT ERROR
0014 337 <VC,,M>,- : VOLUME CHECK
0014 338 <WGE,,M>,- : WRITE GATE ERROR
0014 339 <SPD,,M>,- : SPIN ERROR
0014 340 <SKTO,,M>,- : SEEK TIME OUT
0014 341 <WL,,M>,- : WRITE LOCK
0014 342 <HCE,,M>,- : CURRENT HEAD ERROR
0014 343 <WDE,,M>- : WRITE DATA ERROR
0014 344 >
0014 345 _VIELD RB_MP,0,<- :GET STATUS COMMAND DEFINITIONS
0014 346 <MRK,,M>,- : MARK (ALWAYS 1)
0014 347 <STS,,M>,- : GET STATUS
0014 348 <,1>- : RESERVED
0014 349 <RST,,M>,- : RESET
0014 350 >
0014 351 _VIELD RB_MP,0,<- :RB80 STATUS WORD DEFINITIONS
0014 352 <SEC,5>- : CURRENT RB80 SECTOR
```



```
0014 353 <3>,- ; RESERVED
0014 354 <FLT,M>,- ; DRIVE FAULT
0014 355 <PLGV,M>,- ; PLUG VALID
0014 356 <SKE,M>,- ; SEEK ERROR
0014 357 <ONCY,M>,- ; ON CYLINDER
0014 358 <DRDY,M>,- ; DRIVE READY
0014 359 <WTP,M>,- ; WRITE PROTECT
0014 360 <2>,- ; RESERVED
0014 361 > ; END MPR BIT DEFINITIONS
0014 362
0014 363 $DEF RB_EC1 .BLKL 1 ; ECC POSITION REGISTER (EPOR)
0018 364 -VIELD RB_EC1,0,<- ; START OF EC1 BIT DEFINITIONS
0018 365 <POS,13>,- ; STARTING BIT POSITION OF ECC ERROR
0018 366 <21>- ; RESERVED
0018 367 > ; END EC1 BIT DEFINITIONS
0018 368
0018 369 $DEF RB_EC2 .BLKL 1 ; ECC PATTERN REGISTER (EPAR)
001C 370 -VIELD RB_EC2,0,<- ; START OF EC2 BIT DEFINITIONS
001C 371 <PAT,11>,- ; PATTERN OF ECC ERROR BURST
001C 372 <21>- ; RESERVED
001C 373 > ; END EC2 BIT DEFINITIONS
001C 374
001C 375 $DEF RB_CMD .BLKL 1 ; AUXILLARY COMMAND REGISTER
0020 376 -VIELD RB_CMD,0,<- ; START OF CMD BIT DEFINITIONS
0020 377 <INIT,32>,- ; SUBSYSTEM CLEAR <-- -1
0020 378 > ; END CMD BIT DEFINITIONS
0020 379
0020 380 $DEFEND RB ; END RB730:RB80/RB02 REGISTER DEFS
0000 381
0000 382
0000 383 :
0000 384 :
0000 385 :
00000000 0000 385 F NOP=0*2 ; NO OPERATION
00000004 0000 386 F UNLOAD=2*2 ; GET STATUS/RESET
00000006 0000 387 F SEEK=3*2 ; SEEK CYLINDER
00000006 0000 388 F RECAL=3*2 ; RECALIBRATE (SEEK -1)
00000004 0000 389 F DRVCLR=2*2 ; DRIVE CLEAR (GET STATUS)
00000000 0000 390 F RELEASE=0*2 ; NO OPERATION
00000000 0000 391 F OFFSET=0*2 ; NO OPERATION
00000000 0000 392 F RETCENTER=0*2 ; NO OPERATION
00000004 0000 393 F PACKACK=2*2 ; PACK ACKNOWLEDGE (SET VOLUME VALID)
00000000 0000 394 F STARTSPNDL=0*2 ; NO OPERATION
00000002 0000 395 F WRITECHECK=1*2 ; WRITE CHECK
0000000A 0000 396 F WRITEDATA=5*2 ; WRITE DATA
00000000 0000 397 F WRITEHEAD=0*2 ; WRITE HEADER (WHEN FMT BIT SET)
0000000C 0000 398 F READDATA=6*2 ; READ DATA
00000008 0000 399 F READHEAD=4*2 ; READ HEADER
00000004 0000 400 F GETSTATUS=2*2 ; GET STATUS (DRIVER INTERNAL USE)
00000000 0000 401 F WRITETRACKD=0*2 ; NOP
00000000 0000 402 F READTRACKD=0*2 ; NOP
00000004 0000 403 F AVAILABLE=2*2 ; GET STATUS/RESET
0000 404
```



```
0000 406 .SBTTL STANDARD TABLES
0000 407
0000 408 :
0000 409 : DRIVER PROLOGUE TABLE
0000 410 :
0000 411 : THE DPT DESCRIBES DRIVER PARAMETERS AND I/O DATABASE FIELDS
0000 412 : THAT ARE TO BE INITIALIZED DURING DRIVER LOADING AND RELOADING
0000 413 :
0000 414
0000 415 DPTAB - ;DPT CREATION MACRO
0000 416 END=DQ_END,- ;END OF DRIVER LABEL
0000 417 ADAPTER=UBA,- ;ADAPTER TYPE = UNIBUS
0000 418 FLAGS=DPT$M_SVP,- ;SYSTEM PAGE TABLE ENTRY REQ.
0000 419 MAXUNITS=4,- ;MAXIMUM FOUR DRIVES PER RB730
0000 420 DEFUNITS=4,- ;INTERROGATE FOUR DRIVES
0000 421 DELIVER=DQ_DELIVER,- ;UNIT TEST ROUTINE
0000 422 UCBSIZE=UCB$K_DQ_LEN,- ;LENGTH OF UCB
0000 423 NAME=DQDRIVER ;DRIVER NAME
0038 424
0038 425 DPT_STORE INIT ;START CONTROL BLOCK INIT VALS.
0038 426 DPT_STORE DDB,DDB$$_ACPD,L,<^A\F11\> ;DEFAULT ACP NAME
003F 427 DPT_STORE DDB,DDB$$_ACPD+3,B,DDB$$_PACK ;ACP CLASS
0043 428 DPT_STORE UCB,UCB$$_FIPL,B,8 ;FORK IPL
0047 429 DPT_STORE UCB,UCB$$_DEVCHAR,L,- ;DEVICE CHARACTERISTICS
0047 430 <DEV$M_FOD- ; FILES ORIENTED
0047 431 :DEV$M_DIR- ; DIRECTORY STRUCTURED
0047 432 :DEV$M_AVL- ; AVAILABLE
0047 433 :DEV$M_ELG- ; ERROR LOGGING
0047 434 :DEV$M_SHR- ; SHAREABLE
0047 435 :DEV$M_IDV- ; INPUT DEVICE
0047 436 :DEV$M_ODV- ; OUTPUT DEVICE
0047 437 :DEV$M_RND> ; RANDOM ACCESS
004E 438 DPT_STORE UCB,UCB$$_DEVCHAR2,L,- ; DEVICE CHARACTERISTICS
004E 439 <DEV$M_NNM> ; PREFIX NAME WITH "node$"
0055 440 DPT_STORE UCB,UCB$$_DEVCLASS,B,DQ$$_DISK ; DEVICE CLASS
0059 441 DPT_STORE UCB,UCB$$_DEVBUFSIZ,W,512 ; DEFAULT BUFFER SIZE
005E 442 DPT_STORE UCB,UCB$$_DIPL,B,21 ; DEVICE IPL
0062 443 DPT_STORE UCB,UCB$$_ERTMAX,B,8 ; MAX ERROR RETRY COUNT
0066 444 DPT_STORE UCB,UCB$$_DQ_CURDA,L,-1 ; CURRENT DISK ADDRESS
006D 445
006D 446 DPT_STORE REINIT ; START CONTROL BLOCK RE-INIT
006D 447 DPT_STORE CRB,CRB$$_INTD+4,D,DQ_INT ; INTERRUPT SERV. ROUT. ADDRESS
0072 448 DPT_STORE CRB,CRB$$_INTD+VEC$$_INITIAL,- ; CONTROLLER INIT ADDRESS
0072 449 D,DQ_RB730_INIT ;
0077 450 DPT_STORE CRB,CRB$$_INTD+VEC$$_UNITINIT,- ; UNIT INIT ADDRESS
0077 451 D,DQ_UNIT_INIT ;
007C 452 DPT_STORE DDB,DDB$$_DDT,D,DQ$$_DDT ; DDT ADDRESS
0081 453
0081 454 DPT_STORE END ;END OF INITIALIZATION TABLE
0000 455
0000 456 :
0000 457 : DRIVER DISPATCH TABLE
0000 458 :
0000 459 : THE DDT LISTS ENTRY POINTS FOR DRIVER SUBROUTINES WHICH ARE
0000 460 : CALLED BY THE OPERATING SYSTEM.
0000 461 :
0000 462
```



```
0000 463      DDTAB      -      ;DDT CREATION MACRO
0000 464      DEVNAM=DQ,-      ;NAME OF DEVICE
0000 465      START=DQ STARTIO,-      ;START I/O ROUTINE
0000 466      FUNCTB=DQ FUNCTABLE,-      ;FUNCTION DECISION TABLE
0000 467      CANCEL=0,-      ;CANCEL=NO-OP FOR FILES DEVICE
0000 468      REGDMP=DQ REGDUMP,-      ;REGISTER DUMP ROUTINE
0000 469      DIAGBF=<<RB_NUM_REGS+6+5+3+1>*4>,-      ;BYTES IN DIAG BUFFER
0000 470      ERLGBF=<<<RB_NUM_REGS+6+1>*4>+EMBSL_DV_REGS+1>      ;BYTES IN
0038 471      ;ERROR LOG BUFFER
0038 472      :
0038 473      : DIAGNOSTIC BUFFER SIZE = <<7 RB730 REGISTER LONGWORDS + 6 UCB FIELD LONGWORDS
0038 474      : + 5 IOC$DIAGBUFILL LONGWORDS + 3 BUFFER ALLOCATION
0038 475      : LONGWORDS + 1 LONGWORD FOR # REGISTERS IN DQ_REGDUMP>
0038 476      : * 4 BYTES/LONGWORD>
0038 477      :
0038 478      : ERROR LOG BUFFER SIZE = <<<7 RB730 REGISTER LONGWORDS + 6 UCB FIELD LONGWORDS
0038 479      : + 1 LONGWORD FOR # REGISTERS IN DQ_REGDUMP>
0038 480      : * 4 BYTES/LONGWORD> + BYTES NEEDED FOR ERROR LOGGER
0038 481      : TO SAVE SOFTWARE REGISTERS>
0038 482      :
0038 483      :
0038 484      :
0038 485      : HARDWARE FUNCTION CODE TABLE
0038 486      :
0038 487      : THIS TABLE MERGES THE FUNCTION CODE BITS WITH THE
0038 488      : INTERRUPT ENABLE BIT AND GENERATES THE CASE TABLE
0038 489      : INDEX SYMBOL. THIS IS AN ORDERED TABLE
0038 490      :
0038 491      FTAB:  GENF      F_NOP      ;NO-OP
003C 492      GENF      F_UNLOAD      ;UNLOAD VOLUME (GET STATUS/RESET)
0040 493      GENF      F_SEEK,CRDY      ;SEEK
0044 494      GENF      F_RECAL,CRDY      ;RECALIBRATE
0048 495      GENF      F_DRVCLR      ;DRIVE CLEAR (GET STATUS/RESET)
004C 496      GENF      F_RELEASE      ;RELEASE PORT (NOP)
0050 497      GENF      F_OFFSET      ;OFFSET HEADS (NOP)
0054 498      GENF      F_RETCENTER      ;RETURN HEADS TO CENTERLINE (NOP)
0058 499      GENF      F_PACKACK      ;PACK ACKNOWLEDGE (GET STATUS/RESET)
005C 500      GENF      F_STARTSPNDL      ;START SPINDLE (NOP)
0060 501      GENF      F_WRITECHECK,CRDY      ;WRITE CHECK
0064 502      GENF      F_WRITEDATA,CRDY      ;WRITE DATA
0068 503      GENF      F_READDATA,CRDY      ;READ DATA
006C 504      GENF      F_WRITEHEAD,<CRDY,FMT>      ;WRITE HEADERS
0070 505      GENF      F_READHEAD,CRDY      ;READ HEADERS
0074 506      GENF      F_WRTETRACKD      ;WRITE TRACK DESCRIPTOR (NOP)
0078 507      GENF      F_READTRACKD      ;READ TRACK DESCRIPTOR (NOP)
007C 508      GENF      F_AVAILABLE      ;SET UNIT AVAILABLE (GET STATUS/RESET)
0080 509
```



.SBTTL FUNCTION DECISION TABLES

### FUNCTION DECISION TABLE

THE FDT LISTS VALID FUNCTION CODES, SPECIFIES WHICH CODES ARE BUFFERED, AND DESIGNATES SUBROUTINES TO PERFORM PREPROCESSING FOR PARTICULAR FUNCTIONS.

DQ\_FUNCTABLE:  
FUNCTAB

<NOP,-  
 UNLOAD,-  
 SEEK,-  
 RECAL,-  
 DRVCLR,-  
 PACKACK,-  
 SENSECHAR,-  
 SETCHAR,-  
 SENSEMODE,-  
 SETMODE,-  
 WRITECHECK,-  
 READHEAD,-  
 READBLK,-  
 WRITBLK,-  
 READPBLK,-  
 WRITEPBLK,-  
 READVBLK,-  
 WRITEVBLK,-  
 WRITEHEAD,-  
 AVAILABLE,-  
 ACCESS,-  
 ACPCONTROL,-  
 CREATE,-  
 DEACCESS,-  
 DELETE,-  
 MODIFY,-  
 MOUNT-  
 >

## FUNCTAB

<NOP,-  
 UNLOAD,-  
 SEEK,-  
 RECAL,-  
 DRVCLR,-  
 PACKACK,-  
 AVAILABLE,-  
 SENSECHAR,-  
 SETCHAR,-  
 SENSEMODE,-  
 SETMODE,-  
 ACCESS,-  
 ACPCONTROL,-  
 CREATE,-  
 DEACCESS,-  
 DELETE,-

:LIST LEGAL FUNCTIONS

```

: NO-OP
: UNLOAD
: SEEK
: RECALIBRATE DRIVE
: DRIVE CLEAR
: PACK ACKNOWLEDGE
: SENSE CHARACTERISTICS
: SET CHARACTERISTICS
: SENSE MODE
: SET MODE
: WRITE CHECK
: READ HEADER
: READ LOGICAL BLOCK
: WRITE LOGICAL BLOCK
: READ PHYSICAL BLOCK
: WRITE PHYSICAL BLOCK
: READ VIRTUAL BLOCK
: WRITE VIRTUAL BLOCK
: WRITE DISK HEADERS
: AVAILABLE
: ACCESS FILE / FIND DIRECTORY ENTRY
: ACP CONTROL FUNCTION
: CREATE FILE AND/OR DIRECTORY ENTRY
: DEACCESS FILE
: DELETE FILE AND/OR DIRECTORY ENTRY
: MODIFY FILE ATTRIBUTES
: MOUNT VOLUME

```

## :BUFFERED FUNCTIONS

```

: NO-OP
: UNLOAD
: SEEK
: RECALIBRATE
: DRIVE CLEAR
: PACK ACKNOWLEDGE
: AVAILABLE
: SENSE CHARACTERISTICS
: SET CHARACTERISTICS
: SENSE MODE
: SET MODE
: ACCESS FILE / FIND DIRECTORY ENTRY
: ACP CONTROL FUNCTION
: CREATE FILE AND/OR DIRECTORY ENTRY
: DEACCESS FILE
: DELETE FILE AND/OR DIRECTORY ENTRY

```



0088	568	MODIFY,-	: MODIFY FILE ATTRIBUTES
0088	569	MOUNT-	: MOUNT VOLUME
0088	570	>	
0090	571	FUNCTAB +ACPSREADBLK,-	: READ FUNCTIONS
0090	572	<READHEAD,-	: READ HEADER
0090	573	READLBLK,-	: READ LOGICAL BLOCK
0090	574	READPBLK,-	: READ PHYSICAL BLOCK
0090	575	READVBLK-	: READ VIRTUAL BLOCK
0090	576	>	
009C	577	FUNCTAB +ACPSWRITEBLK,-	: WRITE FUNCTIONS
009C	578	<WRITECHECK,-	: WRITE CHECK
009C	579	WRITEHEAD,-	: WRITE HEADER
009C	580	WRI TELBLK,-	: WRITE LOGICAL BLOCK
009C	581	WRITEPBLK,-	: WRITE PHYSICAL BLOCK
009C	582	WRITEVBLK-	: WRITE VIRTUAL BLOCK
009C	583	>	
00A8	584	FUNCTAB +ACPSACCESS,-	: ACCESS FUNCTIONS
00A8	585	<ACCESS,-	: ACCESS FILE / FIND DIRECTORY ENTRY
00A8	586	CREATE-	: CREATE FILE AND/OR DIRECTORY ENTRY
00A8	587	>	
00B4	588	FUNCTAB +ACPSDEACCESS,-	: DEACCESS FUNCTION
00B4	589	<DEACCESS-	: DEACCESS FILE
00B4	590	>	
00C0	591	FUNCTAB +ACPSMODIFY,-	: MODIFY FUNCTIONS
00C0	592	<ACPCONTROL,-	: ACP CONTROL FUNCTION
00C0	593	DELETE,-	: DELETE FILE AND/OR DIRECTORY ENTRY
00C0	594	MODIFY-	: MODIFY FILE ATTRIBUTES
00C0	595	>	
00CC	596	FUNCTAB +ACPSMOUNT,-	: MOUNT FUNCTION
00CC	597	<MOUNT-	: MOUNT VOLUME
00CC	598	>	
00D8	599	FUNCTAB +EXESLCLDSKVALID,-	: LOCAL DISK VALID FUNCTIONS
00D8	600	<UNLOAD,-	: UNLOAD VOLUME
00D8	601	AVAILABLE,-	: UNIT AVAILABLE
00D8	602	PACKACK-	: PACK ACKNOWLEDGE
00D8	603	>	
00E4	604	FUNCTAB +EXESZEROPARM,-	: ZERO PARAMETER FUNCTIONS
00E4	605	<NOP,-	: NO-OP
00E4	606	UNLOAD,-	: UNLOAD
00E4	607	RECAL,-	: RECALIBRATE
00E4	608	DRVCLR,-	: DRIVE CLEAR
00E4	609	PACKACK,-	: PACK ACKNOWLEDGE
00E4	610	AVAILABLE-	: AVAILABLE
00E4	611	>	
00F0	612	FUNCTAB +EXESONEPARM,-	: ONE PARAMETER FUNCTION
00F0	613	<SEEK-	: SEEK
00F0	614	>	
00FC	615	FUNCTAB +EXESSENSEMODE,-	: SENSE FUNCTIONS
00FC	616	<SENSECHAR,-	: SENSE CHARACTERISTICS
00FC	617	SENSEMODE-	: SENSE MODE
00FC	618	>	
0108	619	FUNCTAB +EXESSETCHAR,-	: SET FUNCTIONS
0108	620	<SETCHAR,-	: SET CHARACTERISTICS
0108	621	SETMODE-	: SET MODE
0108	622	>	
0114	623		



```
0114 625 .SBTTL START I/O ROUTINE
0114 626
0114 627 :++
0114 628
0114 629 DQ_STARTIO - START I/O ROUTINE
0114 630
0114 631 FUNCTIONAL DESCRIPTION:
0114 632
0114 633 THIS FORK PROCESS IS ENTERED FROM THE EXECUTIVE AFTER AN I/O REQUEST
0114 634 PACKET HAS BEEN DEQUEUED, AND PERFORMS THE FOLLOWING:
0114 635
0114 636 - ACTIVATES THE DISK AFTER SETTING UCB FIELDS, OBTAINING
0114 637 UBA AND CONTROLLER RESOURCES, AND SETTING RB730 REGISTERS
0114 638
0114 639 - WAITS FOR AN INTERRUPT
0114 640
0114 641 - REGAINS CONTROL AFTER THE ISR SERVICES THE INTERRUPT, AND
0114 642 - RE-ACTIVATES THE DISK IF THE ORIGINAL FUNCTION
0114 643 IS NOT YET COMPLETE, OR
0114 644 - COMPLETES THE I/O REQUEST BY RELEASING RESOURCES,
0114 645 SETTING STATUS CODES, AND RETURNING TO THE EXEC.
0114 646
0114 647 INPUTS:
0114 648
0114 649 R3 - IRP ADDRESS (I/O REQUEST PACKET)
0114 650 R5 - UCB ADDRESS (UNIT CONTROL BLOCK)
0114 651 IRP$$_MEDIA - PARAMETER LONGWORD (LOGICAL BLOCK NUMBER)
0114 652
0114 653 OUTPUTS:
0114 654
0114 655 R0 - FIRST I/O STATUS LONGWORD: STATUS CODE & BYTES XFERED
0114 656 R1 - SECOND I/O STATUS LONGWORD: 0 FOR DISKS
0114 657
0114 658 THE I/O FUNCTION IS EXECUTED.
0114 659
0114 660 ALL REGISTERS EXCEPT R0-R4 ARE PRESERVED.
0114 661
0114 662 :--
0114 663
0114 664 DQ_STARTIO:
00F2 C5 D0 0114 665 MOVL UCB$$_DQ_CURDA(R5),- ;START I/O OPERATION
00F6 C5 0118 666 UCB$$_DQ_PREVDA(R5) ;SAVE CURRENT DISK ADDRESS
011B 667 ;... FOR ERROR LOGGING
011B 668
011B 669 PREPROCESS UCB FIELDS
011B 670
011B 671
011B 672 PREPROCESS:
011B 673 MOVL IRP$$_MEDIA(R3),- ;ALTERNATE ENTRY NAME
00BC C5 011E 674 UCB$$_MEDIA(R5) ;STORE DISK ADDRESS
0121 675 BDRVTYP RB80,TOS ;BRANCH IF RB80
0127 676 BBS #IRP$$_V_PHYSIO,- ;IF SET - PHYSICAL I/O
0129 677 IRP$$_STS(R3),10$
012C 678 MULL3 #2,UCB$$_MEDIA(R5),R0 ;RB02 HAS 1/2 SECTOR PER BLOCK
0132 679 MOVZBL UCB$$_SECTORS(R5),R2 ;GET NUMBER OF SECTORS PER TRACK
0136 680 CLRL R1 ;CLEAR HIGH PART OF DIVIDEND
00BC C5 50 50 52 7B 0138 681 EDIV R2,R0,R0,UCB$$_MEDIA(R5) ;CALCULATE SECTOR NUMBER AND STORE
```



```
51 52 45 A5 9A 013F 682      MOVZBL UCB$B_TRACKS(R5),R2      ;GET NUMBER OF TRACKS PER CYLINDER
50 50 52 7B 0143 683      EDIV      R2,R0,R0,R1      ;CALCULATE TRACK AND CYLINDER
00BD C5 51 90 0148 684      MOV      R1,UCB$L_MEDIA+1(R5)    ;STORE TRACK NUMBER
00BE C5 50 B0 014D 685      MOVW     R0,UCB$L_MEDIA+2(R5)    ;STORE CYLINDER NUMBER
                                0152 686
                                0152 687 10$:      MOV      UCB$B_ERTMAX(R5),-      ;INITIALIZE ERROR RETRY COUNT
                                0156 688      UCB$B_ERTCNT(R5)
00C0 C5 7E A5 B0 0159 689      MOVW     UCB$W_BCNT(R5),UCB$W_BCR(R5)    ;INITIALIZE REMAINING BYTE COUNT
009A C5 20 A3 B0 015F 690      MOVW     IRP$W_FUNC(R3),UCB$W_FUNC(R5) ;SAVE FUNCTION CODE AND MODIFIERS
                                0165 691      EXTZV     #IRP$V_FCODE,-      ;EXTRACT I/O FUNCTION CODE
51 20 A3 06 90 0167 692      #IRP$S_FCODE,IRP$W_FUNC(R3),R1      ;
0092 C5 51 91 016B 693      MOV      R1,UCB$B_FEX(R5)      ;STORE FUNCTION DISPATCH INDEX
                                0170 694      CMPB      #10$_SEER,R1      ;SEEK FUNCTION?
                                0173 695      BNEQ      30$      ;IF NEQ - NO
00BC C5 10 78 0175 696      ASHL      #16,UCB$L_MEDIA(R5),-      ;SHIFT CYLINDER ADDRESS
                                017A 697      UCB$L_MEDIA(R5)      ;...INTO HIGH WORD
                                017D 698 30$:      BICW      #UCB$M_DIAGBUF-      ;CLEAR DIAGNOSTIC BUFFER PRESENT
                                017E 699      !UCB$M_ECC,-      ;...AND ECC CORRECTION MADE FLAGS
68 A5 03 017E 700      UCB$W_DEVSTS(R5)      ;...IN DEVICE STATUS WORD
                                0181 701      BBC      #IRP$V_DIAGBUF,-      ;IF CLR - NO DIAG BUFFER
                                0183 702      IRP$W_STS(R3),FDISPATCH
68 A5 02 A8 0186 703      BISW      #UCB$M_DIAGBUF,UCB$W_DEVSTS(R5) ;SET DIAG BUFFER PRESENT
                                018A 704
                                018A 705
                                018A 706
                                018A 707      CENTRAL FUNCTION DISPATCH
                                018A 708
                                018A 709
                                018A 710 FDISPATCH:
53 58 A5 D0 018A 711      MOVL      UCB$L_IRP(R5),R3      ;FUNCTION DISPATCH
10 2A A3 E0 018E 712      BBS      #IRP$V_PHYSIO,-      ;GET IRP ADDRESS
                                0190 713      IRP$W_STS(R3),10$      ;IF SET - PHYSICAL I/O FUNCTION
0B 64 A5 E0 0193 714      BBS      #UCB$V_VALID,-      ;IF SET - VOLUME SOFTWARE VALID
50 0254 8F 3C 0195 715      UCB$W_STS(R5),10$      ;
7E A5 B4 0198 716      MOVZWL     #SS$VOLINV,R0      ;SET VOLUME INVALID STATUS
017E 31 01A0 717      CLRW      UCB$W_BCNT(R5)      ;SET ZERO BYTES TRANSFERRED
                                01A3 718      BRW      FUNCXT      ;AND RETURN TO CALLER
                                01A3 719
53 00C9 C5 94 01A3 720 10$:      CLRB      UCB$B_DQ_FLAGS(R5)      ;CLEAR LOCAL FLAGS
0092 C5 9A 01A7 721      MOVZBL     UCB$B_FEX(R5),R3      ;GET FUNCTION DISPATCH INDEX
                                01AC 722      CASE      R3,-      ;DISPATCH TO FUNCTION HANDLING ROUTINE
                                01AC 723      NOP      -      ;NO OPERATION
                                01AC 724      UNLOAD,-      ;UNLOAD
                                01AC 725      SEEK,-      ;SEEK
                                01AC 726      RECAL,-      ;RECALIBRATE
                                01AC 727      DRVCLR,-      ;DRIVE CLEAR
                                01AC 728      RELEASE,-      ;RELEASE
                                01AC 729      OFFSET,-      ;OFFSET HEADS
                                01AC 730      RETCENTER,-      ;RETURN TO CENTER
                                01AC 731      PACKACK,-      ;PACKACK
                                01AC 732      STARTSPNDL,-      ;START SPINDLE
                                01AC 733      WRITECHECK,-      ;WRITE CHECK
                                01AC 734      WRITEDATA,-      ;WRITE DATA
                                01AC 735      READDATA,-      ;READ DATA
                                01AC 736      WRITEHEAD,-      ;WRITE HEADER
                                01AC 737      READHEAD,-      ;READ HEADER
                                01AC 738      WRITETRACKD,-      ;WRITE TRACK DESCRIPTOR
```



```
01AC 739 READTRACKD,- ; READ TRACK DESCRIPTOR
01AC 740 AVAILABLE,- ; UNIT AVAILABLE
01AC 741 >
01D4 742
01D4 743 :
01D4 744 : IOS UNLOAD AND IOS AVAILABLE INDICATE THE UNIT IS NOT MOUNTED
01D4 745 : SO WE CLEAR SOFTWARE VOLUME VALID. IOS PACKACK INDICATES THAT
01D4 746 : SOFTWARE IS READY TO MOUNT OR ACCESS VOLUME SO WE SET SOFTWARE
01D4 747 : VALID. ON PACKACKS'S WE FOLLOW THIS WITH A GET STATUS AND RESET.
01D4 748 : IF THE OBTAINED STATUS INDICATES THAT THE DRIVE IS NOT READY
01D4 749 : THEN VOLUME VALID WILL BE CLEARED.
01D4 750 :
01D4 751
01D4 752 PACKACK: ;PACK ACKNOWLEDGE
0E 64 0B E2 01D4 753 BBSS #UCBSV_VALID,- ;SET SOFTWARE VOLUME VALID
0E 64 A5 01D6 754 UCB$W_STS(R5),NOP ;...
43 11 01D9 755 EXFUNCL RETRYERR,F_DRVCLR ;GET STS AND RESET, RETRY ERRORS
01E0 756 BRB NORMAL ;SUCCESSFUL - EXIT WITH NORMAL STATUS
01E2 757
01E2 758 UNLOAD: ;UNLOAD
01E2 759 AVAILABLE: ;UNIT AVAILABLE
00 64 0B E5 01E2 760 BBCC #UCBSV_VALID,- ;CLEAR SOFTWARE VALID
00 64 A5 01E4 761 UCB$W_STS(R5),NOP ;...
01E7 762 NOP: ;NO-OP
01E7 763 RELEASE: ;RELEASE PORT (NOP)
01E7 764 OFFSET: ;OFFSET HEADS (NOP)
01E7 765 RETCENTER: ;RETURN TO CENTERLINE (NOP)
01E7 766 STARTSPNDL: ;START SPINDLE (NOP)
01E7 767 WRITETRACKD: ;WRITE TRACK DESCRIPTOR (NOP)
01E7 768 READTRACKD: ;READ TRACK DESCRIPTOR (NOP)
35 11 01E7 769 EXFUNCL RETRYERR,F_NOP ;EXECUTE A HARDWARE NOP, RETRY ERRORS
01EE 770 BRB NORMAL ;SUCCESSFUL - EXIT WITH NORMAL STATUS
01F0 771
01F0 772 SEEK: ;SEEK
01F0 773 RECAL: ;RECALIBRATE
01F0 774 DRVCLR: ;DRIVE CLEAR (GET STATUS & RESET)
01F0 775 WRITEHEAD: ;WRITE HEADERS (AND DATA)
2F 11 01F0 776 EXFUNCL RETRYERR ;EXECUTE FUNCTION - RETRY IF FAILURE
01F4 777 BRB NORMAL ;SUCCESSFUL - EXIT WITH NORMAL STATUS
01F6 778
01F6 779 WRITECHECK: ;WRITE CHECK
01F6 780 READHEAD: ;READ HEADER
4000 8F AA 01F6 781 BICW #IOSM_DATACHECK,- ;CLEAR DATA CHECK REQUEST-
009A C5 01FA 782 UCB$W_FUNC(R5) ;TO PREVENT EXTRA WRITE CHECK
01FD 783
01FD 784 WRITEDATA: ;WRITE DATA
01FD 785 READDATA: ;READ DATA
00BE C5 B1 01FD 786 CMPW UCB$SL_MEDIA+2(R5),- ;NEW CYLINDER?
00F4 C5 0201 787 UCB$SL_DQ_CURDA+2(R5) ;...
OF 12 0204 788 BNEQ 20$ ;BRANCH IF SO
0206 789 BDRVTYP RB80,TRANSFER ;BRANCH IF RB80
00BD C5 91 020C 790 CMPB UCB$SL_MEDIA+1(R5),- ;OR NEW TRACK? (MUST DO SEEK TO
00F3 C5 0210 791 UCB$SL_DQ_CURDA+1(R5) ;...SELECT HEAD ON RB02)
OC 13 0213 792 BEQL TRANSFER ;BRANCH IF NO SEEK REQUIRED
53 0092 C5 9A 0215 793 20$: EXFUNCL RETRYERR,F SEEK ;EXECUTE EXPLICIT SEEK - RETRY IF ERROR
021C 794 MOVZBL UCB$B_FEX(R5),R3 ;RESTORE FUNCTION DISPATCH INDEX
0221 795
```

```
0221 796 :  
0221 797 : DRIVE HAS BEEN POSITIONED -- NOW EXECUTE THE TRANSFER  
0221 798 :  
0221 799 :  
0221 800 TRANSFER:  
0221 801 EXFUNCL CHECKECC ;EXECUTE TRANSFER FUNCTION  
0225 802  
0225 803  
0225 804 :  
0225 805 : OPERATON COMPLETION  
0225 806 :  
0225 807 :  
0225 808 NORMAL :  
50 0639 8F 3C 0225 809 MOVZWL #SS$ WASECC,R0 ;SUCCESSFUL OPERATION COMPLETE  
03 68 A5 E0 022A 810 BBS #UCBSV ECC,- ;ASSUME CORRECTED ECC ERROR  
50 01 3C 022C 811 UCBSW DEVSTS(R5),10$ ;BRANCH IF CORRECTED ECC  
00EC 31 022F 812 MOVZWL #SS$ NORMAL,R0 ;...ERROR OCCURED  
0232 813 10$: BRW FUNCXT ;SET NORMAL COMPLETION STATUS  
0235 814 ;FUNCTION EXIT
```



```
.SBTTL RETRIABLE ERROR ANALYSIS

A RETRIABLE ERROR HAS OCCURED ON A TRANSFER
CHECK TO SEE IF ECC CORRECTION CAN BE APPLIED

R1 - CSR AT TIME OF ERROR
R2 - MPR OF GET STATUS FOLLOWING ERROR

CHECKECC:
06 51 01 ED 0235 816
03 03 0235 817
06E 12 0235 818
03 03 0235 819
03 03 0235 820
03 03 0235 821
03 03 0235 822
03 03 0235 823
03 03 0235 824
03 03 0235 825
03 03 0237 826
03 03 023A 827
03 03 023A 828
03 03 023C 829
03 03 0242 830
03 03 0243 831
03 03 0243 832
03 03 0243 833
03 03 0243 834
03 03 0247 835
03 03 0249 836
03 03 024D 837
03 03 024F 838
03 03 0250 839
03 03 0252 840
03 03 0252 841
03 03 0254 842
03 03 0254 843
03 03 0254 844
03 03 0254 845
03 03 0254 846
03 03 0254 847
03 03 0254 848
03 03 0254 849
03 03 0254 850
03 03 0254 851
03 03 0258 852
03 03 025B 853
03 03 025F 854
03 03 025F 855
03 03 0264 856
03 03 026B 857
03 03 026D 858
03 03 0272 859
03 03 0274 860
03 03 0277 861
03 03 027E 862
03 03 027E 863
03 03 0282 864
03 03 0282 865
03 03 0284 866
03 03 0286 867
03 03 028D 868
03 03 028F 869
03 03 028F 870
03 03 0291 871
03 03 0291 872

CMPZV #RB_CS_V_FCODE,- :WAS THIS A READ DATA OPERATION?
#RB_CS_S_FCODE,R1,- :...
#<F_READDATA @ -1> :...
BNEQ RETRYERR :BRANCH IF NOT
BDRVTYP RB02,RETRYERR :BRANCH IF RB02
BITW #RB_CS_M_DE- :DRIVE ERROR
!RB_CS_M_NXM- :...OR NON EXISTENT MEMORY
!RB_CS_M_DLT- :...OR DATA LATE
!RB_CS_M_OPI,- :...OR OPERATION INCOMPLETE (HDR CRC)
R1 :...?
BNEQ RETRYERR :BRANCH IF SO
BBC #RB_CS_V_DCK,R1,RETRYERR :BRANCH IF NOT A DATACHECK
CMPZV #RB_CS_V_ECS,- :COMPARE ECC STATUS BITS (START)
#RB_CS_S_ECS,- :... (SIZE)
R1,- :... (FROM)
#^B11 :...TO BINARY 11 (BOTH SET)
BNEQ RETRYERR :BRANCH IF NOT CORRECTABLE

THIS IS A CORRECTABLE ECC ERROR -- IF IT IS A SINGLE BIT
ERROR THEN APPLY THE ECC CORRECTION. IF IT IS A MULTIPLE
BIT ERROR THEN FINISH PROCESSING THE GOOD BLOCKS IN FRONT
OF THE ERROR, REREAD THE ERROR BLOCK UNTIL THE ERROR IS
CORRECTED OR THE RETRY COUNT IS ZERO. IF THE RETRY COUNT
REACHES ZERO THEN APPLY CORRECTION.

BICW #RB_CS_M_CE,- :CLEAR COMBINED ERROR IN
UCBSL_DQ_CS(R5) :...CASE WE CONTINUE
MOVZWL UCBSW_BCNT(R5),R0 :FETCH ORIGINAL XFER COUNT (AS CORRECTED
:BY RETREG BUT INCLUDING ECC BLOCK)
ADDL UCBSL_DQ_BC(R5),R0 :COMPUTE BYTES TRANSFERED
SUBL #^X200,R0 :BACKUP TO LAST GOOD BLOCK
BLSS RETRYERR :NEGATIVE, SOMETHING WRONG, TRY AGAIN
BITW #^X1FF,R0 :WHOLE BLOCKS TRANSFERED?
BNEQ RETRYERR :NO, SOMETHING WRONG, TRY AGAIN
MOVQ R2,-(SP) :SAVE WORK REGISTERS
FFS #0,#11,UCBSW_EC2(R5),R2 :FIND THE FIRST ERROR BIT SET IN THE
: ECC PATTERN REGISTER
SUBL3 R2,#10,R3 :GET THE NUMBER OF SET ERROR BITS IN
: THE REMAINDER OF THE PATTERN
BLEQ 10$ :BRANCH IF NO OTHER BITS SET
INCL R2 :POINT TO NEXT BIT IN PATTERN
EXTZV R2,R3,UCBSW_EC2(R5),R2 :IS THERE MORE THAN ONE ERROR BIT SET?
POPR #^M<R3,R2> :RESTORE WORK REGISTERS WITHOUT
: AFFECTING FLAGS
BLEQU APPLY_ECC :IF ONLY ONE ERROR BIT SET, THEN APPLY
: ECC CORRECTION
BISB #UCBSM_DQ_ECC_DEFER,- :SIGNAL ECC CORRECTION DEFERRED
```

```
00C9 C5      0293 873      UCBSB_DQ_FLAGS(R5)      ;
7E A5 0200 8F A2 0296 874      SUBW #^X200,UCBSW_BCNT(R5) ;SHORTEN XFER BY ONE PAGE TO EXCLUDE ECC
                                0296 875      ;...BLOCK FROM NUMBER OF BYTES
                                029C 876      ;...TRANSFERRED
                                029C 877      ;BRANCH IF NO GOOD DATA WAS TRANSFERRED
                                0C 13 029C 878      BEQL RETRYERR ;AND ATTEMPT TO RETRY
                                034D 31 029E 879      BRW WRITECHK ;OTHERWISE, BRANCH AND PERFORM A
                                02A1 881      ;...WRITECHECK
                                02A1 882
                                02A1 883 APPLY_ECC:
00000000'GF 16 02A1 884      JSB G^IOCSAPPLYECC ;APPLY ECC CORRECTION
035B 31 02A7 885      BRW UPDATE ;CONTINUE TRANSFER BUT SUPPRESS
                                02AA 886      ;...WRITECHECK
                                02AA 887
                                02AA 888
                                02AA 889      ; A RETRIABLE ERROR HAS OCCURED
                                02AA 890
                                02AA 891      R1 - CSR AT TIME OF ERROR
                                02AA 892      R2 - MPR OF GET STATUS FOLLOWING ERROR
                                02AA 893
                                02AA 894
                                02AA 895 RETRYERR:
0080 C5 97 02AA 896      DECB UCBSB_ERTCNT(R5) ;ANY RETRIES LEFT?
08 14 02AE 897      BGTR RESETDRIVE ;IF GTR - YES
02 E4 02B0 898      BBSC #UCBSV_DQ_ECC_DEFER,- ;CORRECT THE ERROR WITH ECC IF WE CAN
00C9 C5 02B2 899      UCBSB_DQ_FLAGS(R5),-
EB 0A 11 02B5 900      APPLY_ECC
                                02B6 901      BRB FATALERR ;OTHERWISE, FATAL ERROR
                                02B8 902
                                02B8 903      ; ATTEMPT TO RESET STUBBORN DRIVE ERRORS BEFORE EXECUTING THE FUNCTION AGAIN
                                02B8 904
                                02B8 905
                                02B8 906
                                02B8 907 RESETDRIVE:
FEC8 31 02B8 908      EXFUNCL RETRYERR,F_RECAL ;RECALIBRATE THE DRIVE
                                02BF 909      BRW FDISPATCH ;RETRY FUNCTION
                                02C2 910
```



```
02C2 912 .SBTTL FATAL ERROR ANALYSIS
02C2 913
02C2 914 :
02C2 915 A FATAL ERROR HAS OCCURED
02C2 916
02C2 917 R1 - CSR AT TIME OF ERROR OR
02C2 918 R2 - MPR OF GET STATUS FOLLOWING ERROR
02C2 919
02C2 920 FATALERR: UNRECOVERABLE ERROR
02C2 921 ASSUME RB_MP_V_WL EQ RB_MP_V_WTP :ASSUME RB02 AND RB80 USE
02C2 922 :SAME BIT FOR WRITE LOCK
02C2 923
02C2 924 BBC #RB_MP_V_WL,R2,30$ :BRANCH IF DRIVE IS NOT WRITELOCKED
50 13 52 0D E1 02C6 925 MOVZWL #SS$_WRITLCK,R0 :ASSUME WRITELOCK ERROR STATUS
025C 8F 3C 02CB 926 CMPB #CDF_WRTEDATA,- :WAS THIS A WRITE DATA OPERATION?
0093 C5 91 02CD 927 UCBS$ CEX(R5) :
4F 13 02D0 928 BEQL FUNCXT :BRANCH IF SO
0D 91 02D2 929 CMPB #CDF_WRITEHEAD,- :WAS THIS A WRITE HEADER OPERATION?
0093 C5 02D4 930 UCBS$ CEX(R5) :
48 13 02D7 931 BEQL FUNCXT :BRANCH IF SO
50 0254 8F 3C 02D9 932 30$: MOVZWL #SS$_VOLINV,R0 :ASSUME VOLUME INVALID
02DE 934 BDRVTYP RB02,50$ :BRANCH IF RB02
02E4 935
02E4 936 :
02E4 937 :
02E4 938 :
11 11 02E4 939 BRB 70$ :CONTINUE IN COMMON
02E6 940
02E6 941
02E6 942 :
02E6 943 :
02E6 944 :
02E6 945 :
02E6 946 50$: BBS #RB_MP_V_VC,R2,FUNCXT :IF SET - VOLUME INVALID
50 37 52 09 E0 02EA 947 MOVZWL #SS$_WRITLCK,R0 :ASSUME WRITE LOCK ERROR STATUS
025C 8F 3C 02EF 948 BBC #RB_MP_V_WL,R2,70$ :IF CLR - VOLUME NOT WRITE LOCKED
04 52 0D E1 02F3 949 BBS #RB_MP_V_WGE,R2,FUNCXT :IF SET - WRITE GATE ERROR
2A 52 0A E0 02F7 950
02F7 951 70$: MOVZWL #SS$_DATACHECK,R0 :ASSUME DATA CHECK ERROR STATUS
50 005C 8F 3C 02FC 952 BBC #UCBS$V_DQ_DIP,- :BRANCH IF NO DATA CHECK IN PROGRESS
01 E1 02FE 953 UCBS$V_DQ_FLAGS(R5),80$ :
08 00C9 C5 0302 954 BBC #RB_CS_V_OPI,R1,80$ :DATA CHECK INDICATED BY OPI AND
04 51 0A E1 0306 955 BBS #RB_CS_V_DCK,R1,FUNCXT :... DATA CHECK SET
17 51 0B E0 030A 956
030A 957 80$: MOVZWL #SS$_PARITY,R0 :ASSUME PARITY ERROR STATUS
50 01F4 8F 3C 030F 958 BBS #RB_CS_V_DCK,R1,FUNCXT :IF SET - CRC ERROR
0E 51 0B E0 0313 959
0313 960 90$: MOVZWL #SS$_DRVERR,R0 :ASSUME DRIVE ERROR STATUS
50 008C 8F 3C 0318 961 BBS #RB_CS_V_DE,R1,FUNCXT :IF SET - DRIVE ERROR
05 51 0E E0 031C 962
031C 963 MOVZWL #SS$_CTRLERR,R0 :ASSUME CONTROLLER ERROR STATUS
50 0054 8F 3C 0321 964
```

```
0321 966 .SBTTL FUNCTION COMPLETION
0321 967
0321 968 :
0321 969 :
0321 970 :
0321 971 :
0321 972 :
0321 973 FUNCXT: ;FUNCTION EXIT
0321 974 PUSHL R0 ;SAVE FINAL REQUEST STATUS
00000000 GF DD 0321 974 JSB G*IOCS$DIAGBUFILL ;FILL DIAGNOSTIC BUFFER IF PRESENT
0092 C5 0A 91 0323 975 CMPB #CDF_WRITECHECK,UCB$B_FEX(R5) ;DRIVE RELATED FUNCTION?
53 58 A5 D0 032E 977 BGTRU 50$ ;IF GTRU - YES
0A 6E E8 0330 978 MOVL UCB$L_IRP(R5),R3 ;RETRIEVE ADDRESS OF IRP
0334 979 BLBS (SP),T0$ ;BRANCH IF XFER SUCCESSFUL
0337 980 :
0337 981 :THE TRANSFER ENDED IN AN ERROR -- COMPUTE BYTES SUCCESSFULLY TRANSFERRED.
0337 982 :THE BYTE-COUNT-REMAINING REPORTED BY THE DRIVE CANNOT BE TRUSTED IF A
0337 983 :DRIVE ERROR OCCURED.
0337 984 :
06 51 0E E0 0337 985 BBS #RB CS V DE,R1,10$ ;IGNORE FINAL SEGMENT IF DRIVE ERROR
7E A5 A2 033B 986 SUBW UCB$W_BCNT(R5),- ;UPDATE BCR WITH PARTIAL XFER COUNT
00C0 C5 033E 987 UCB$W_BCR(R5) ;...FROM FINAL SEGMENT
00C0 C5 A3 0341 988 10$: SUBW3 UCB$W_BCR(R5),- ;CALCULATE BYTES TRANSFERRED
02 AE 32 A3 0345 989 IRP$W_BCNT(R3),2(SP) ;...
0349 990
0349 991 50$: RELCHAN ;RELEASE CHANNEL IF OWNED
51 D4 034F 992 CLRL R1 ;CLEAR SECOND STATUS LONGWORD
50 BED0 0351 993 POPL R0 ;RETRIEVE FINAL REQUEST STATUS
0354 994 REQCOM ;COMPLETE REQUEST
035A 995
```



```
035A 997      .SBTTL  HARDWARE FUNCTION DISPATCH
035A 998      :
035A 999      : FEXL - RB730 HARDWARE FUNCTION EXECUTION
035A 1000     :
035A 1001     : THIS ROUTINE IS CALLED VIA A BSB WITH A BYTE IMMEDIATELY FOLLOWING THAT
035A 1002     : SPECIFIES THE ADDRESS OF AN ERROR ROUTINE. ALL DATA IS ASSUMED TO HAVE BEEN
035A 1003     : SET UP IN THE UCB BEFORE THE CALL. THE APPROPRIATE PARAMETERS ARE LOADED
035A 1004     : INTO DEVICE REGISTERS AND THE FUNCTION IS INITIATED. THE RETURN ADDRESS
035A 1005     : IS STORED IN THE UCB AND A WAITFOR INTERRUPT IS EXECUTED. WHEN THE
035A 1006     : INTERRUPT OCCURS, CONTROL IS RETURNED TO THE CALLER.
035A 1007     :
035A 1008     : INPUTS:
035A 1009     :
035A 1010     : R3 = FUNCTION TABLE DISPATCH INDEX
035A 1011     : R5 = DEVICE UNIT UCB ADDRESS
035A 1012     :
035A 1013     : 00(SP) = RETURN ADDRESS OF CALLER
035A 1014     : 04(SP) = RETURN ADDRESS OF CALLER'S CALLER
035A 1015     :
035A 1016     : IMMEDIATELY FOLLOWING INLINE AT THE CALL SITE IS A BYTE WHICH CONTAINS
035A 1017     : A BRANCH DESTINATION TO AN ERROR RETRY ROUTINE.
035A 1018     :
035A 1019     : OUTPUTS:
035A 1020     :
035A 1021     : THERE ARE FOUR EXITS FROM THIS ROUTINE:
035A 1022     :
035A 1023     : 1. SPECIAL CONDITION - THIS EXIT IS TAKEN IF A POWER FAILURE OCCURS
035A 1024     : OR THE OPERATION TIMES OUT. IT IS A JUMP TO THE APPROPRIATE
035A 1025     : ERROR ROUTINE. NO DEVICE REGISTERS ARE SAVED.
035A 1026     :
035A 1027     : 2. FATAL ERROR - THIS EXIT IS TAKEN IF A FATAL CONTROLLER OR DRIVE
035A 1028     : ERROR OCCURS OR IF ANY ERROR OCCURS AND ERROR RETRY IS EITHER
035A 1029     : INHIBITED OR EXHAUSTED. IT IS A JUMP TO THE FATAL ERROR EXIT
035A 1030     : ROUTINE. ALL DEVICE REGISTERS ARE SAVED.
035A 1031     :
035A 1032     : 3. RETRIABLE ERROR - THIS EXIT IS TAKEN IF A RETRIABLE CONTROLLER
035A 1033     : OR DRIVE ERROR OCCURS AND ERROR RETRY IS NEITHER INHIBITED
035A 1034     : NOR EXHAUSTED. IT CONSISTS OF TAKING THE ERROR BRANCH EXIT
035A 1035     : SPECIFIED AT THE CALL SITE. ALL DEVICE REGISTERS ARE SAVED.
035A 1036     :
035A 1037     : 4. SUCCESSFUL OPERATION - THIS EXIT IS TAKEN IF NO ERRORS OCCUR
035A 1038     : DURING THE OPERATION. IT CONSISTS OF A RETURN INLINE.
035A 1039     : ONLY THE CSR IS SAVED.
035A 1040     :
035A 1041     : IN ALL CASES IF AN ERROR OCCURS, AN ATTEMPT IS MADE TO LOG THE ERROR.
035A 1042     :
035A 1043     :
035A 1044     :
```



```
0093 009C C5 8ED0 035A 1046 FEXL:
50 C5 53 90 035A 1047 POPL UCBSL_DPC(R5)
51 24 A5 D0 035F 1048 MOV B R3,UCBSB_CEX(R5)
04 A1 2C A0 D0 0364 1049 MOV L UCBSL_CRB(R5),R0
54 55 D1 0368 1050 MOV L CRBSL_INTD+VECSL_IDB(R0)
05 12 036C 1051 CMPL R5,IDBSL_OWNER(RT)
61 D0 0370 1052 BNEQ 10$
06 11 0372 1053 MOV L IDBSL_CSR(R1),R4
0375 1054 BRB 20$
0377 1055 10$: REQPCN
037D 1056 20$: CASE R3,<-
037D 1057 IMMED,-
037D 1058 IMMED,-
037D 1059 IMMED,-
037D 1060 POSIT,-
037D 1061 RECALB,-
037D 1062 DRCLR,-
037D 1063 IMMED,-
037D 1064 IMMED,-
037D 1065 IMMED,-
037D 1066 DRCLR,-
037D 1067 IMMED,-
037D 1068 >
0142 31 0395 1069 BRW XFER
```

:FUNCTION EXECUTOR  
:SAVE DRIVER PC VALUE  
:SAVE CASE INDEX  
:GET ADDRESS OF PRIMARY CRB  
:GET ADDRESS OF IDB  
:DOES THIS PROCESS OWN CHANNEL?  
:IF NEQ - NO  
:SET ASSIGNED CHANNEL CSR ADDRESS  
:REQUEST CHANNEL (RETURNS R4 = CSR ADR)  
:DISPATCH TO PROPER FUNCTION ROUTINE  
:NO OPERATION  
:UNLOAD VOLUME (NOP)  
:SEEK CYLINDER  
:RECALIBRATE  
:DRIVE CLEAR (GET STATUS & RESET)  
:RELEASE DRIVE (NOP)  
:OFFSET HEADS (NOP)  
:RETURN TO CENTERLINE (NOP)  
:PACK ACKNOWLEDGE  
:START SPINDLE (NOP)  
:TRANSFER FUNCTION



```
0398 1071 .SBTTL IMMEDIATE FUNCTION EXECUTION
0398 1072 :
0398 1073 : IMMEDIATE FUNCTION EXECUTION
0398 1074 :
0398 1075 : FUNCTIONS INCLUDE:
0398 1076 :
0398 1077 : NO OPERATION,
0398 1078 : DRIVE CLEAR, AND
0398 1079 : PACK ACKNOWLEDGE
0398 1080 :
0398 1081 : INPUTS:
0398 1082 : R3 - CASE INDEX
0398 1083 : R4 - CSR ADDRESS
0398 1084 : R5 - UCB ADDRESS
0398 1085 :
0398 1086 : FUNCTIONAL DESCRIPTION:
0398 1087 :
0398 1088 : INTERRUPTS ARE LOCKED OUT, THE APPROPRIATE FUNCTION IS INITIATED WITH
0398 1089 : INTERRUPT ENABLE, AND A WAITFOR INTERRUPT AND KEEP CHANNEL IS EXECUTED.
0398 1090 :
0398 1091 : THE RESET COMMAND DOES NOT AFFECT AN R80 SO A RECALIBRATE, WHICH CLEARS
0398 1092 : ERRORS, IS PERFORMED INSTEAD
0398 1093 :
0398 1094 :
0398 1095 DRCLR: ;GET STATUS AND RESET
0398 1096 BDRVTYP RB02,10$ ;BRANCH IF RB02
0093 53 03 9A 039E 1097 MOVZBL #CDF_RECAL,R3 ;SET FUNCTION AS RECALIBRATE
0398 1098 MOVVB R3,UCB$B_CEX(R5) ;SAVE CASE INDEX
0398 1099 BRB RECALB ;AND BRANCH TO EXECUTION
0398 1100
0398 1101 10$: MOVL #RB_MP_M_STS- ;GET STATUS AND
0398 1102 !RB_MP_M_RST- ;...RESET DRIVE
0398 1103 !RB_MP_M_MRK,- ;...INDICATE GET STATUS COMMAND PRESENT
50 08 03 11 03A9 1104 R0 ;...IN R0
0398 1105 BRB EX_IMED ;EXECUTE THE FUNCTION
0398 1106
0398 1107 IMMED: ;GET STATUS
0398 1108 MOVL #RB_MP_M_STS- ;GET STATUS AND
0398 1109 !RB_MP_M_MRK,- ;...INDICATE GET STATUS COMMAND PRESENT
50 03 03AE 1110 R0 ;...IN R0
0398 1111
0398 1112 EX_IMED: ;EXECUTE IMMEDIATE FUNCTION
52 52 FC83 CF43 DO 03B0 1113 MOVL FTAB[R3],R2 ;FETCH FUNCTION CODE AND MODIFIERS
02 02 08 54 A5 FO 03B6 1114 INSV UCB$W_UNIT(R5),#8,#2,R2 ;MERGE UNIT NUMBER
0398 1115 SAVIPL ;SAVE CURRENT IPL ON STACK
0398 1116 CKPWR 10$ ;RAISE IPL AND CHECK FOR POWERFAIL
10 A4 50 DO 03CC 1117 MOVL R0,RB_MP(R4) ;PREPARE FOR GETSTATUS OR RESET
64 52 DO 03D0 1118 MOVL R2,RB_CS(R4) ;INITIATE FUNCTION
0398 1119 WFIKPC RETREG,#10 ;WAITFOR INTERRUPT
0398 1120 IOFORK ;CREATE FORK PROCESS
02CA 31 03E3 1121 10$: BRW RETREG
0398 1122
```



```
03E6 1124 .SBTTL RECALIBRATE FUNCTION EXECUTION
03E6 1125 :
03E6 1126 : RECALIBRATE FUNCTION EXECUTION
03E6 1127 :
03E6 1128 : FUNCTIONS INCLUDE:
03E6 1129 :
03E6 1130 : RECALIBRATE
03E6 1131 :
03E6 1132 : INPUTS:
03E6 1133 : R3 - CASE INDEX
03E6 1134 : R4 - DEVICE CSR ADDRESS
03E6 1135 : R5 - UCB ADDRESS
03E6 1136 :
03E6 1137 : FUNCTIONAL DESCRIPTION:
03E6 1138 :
03E6 1139 : FOR AN RB80, A RECALIBRATE IS PERFORMED. THE HEADS SEEK TO A KNOWN
03E6 1140 : POSITION TO RESET THE DRIVES PHYSICAL POSITIONING LOGIC. THIS IS
03E6 1141 : SIGNED BY REQUESTING A SEEK TO -1
03E6 1142 :
03E6 1143 : FOR AN RB02, A READ HEADER IS PERFORMED. THE RB02 DOES NOT SUPPORT
03E6 1144 : A RECALIBRATE FUNCTION. HOWEVER, THE SUPPORT MICROCODE FOR THE RB02
03E6 1145 : KEEPS AN INTERNAL RECORD OF THE CURRENT DISK POSITION. WHENEVER A
03E6 1146 : READ HEADER IS PERFORMED, IT UPDATES THAT POSITION WITH THE VALUE
03E6 1147 : FOUND IN THE HEADER.
03E6 1148 :
03E6 1149 :
03E6 1150 RECALB:
03E6 1151 MNEGL #1,UCBSL_DQ_CURDA(R5) ;RECALIBRATE FUNCTION
03EB 1152 BDRVTYP RB02,50$ ;DISABLE SEEK OPTIMIZATION
03F1 1153 GETUNIT ;BRANCH IF RB02
03F9 1154 DSBINT UCBSB_DIPL(R5) ;GET UNIT NUMBER IN R2
64 52 FC33 CF43 C9 0400 1155 BISL3 FTAB[R3],R2,RB_CS(R4) ;SAVE IPL AND LOCK OUT DEVICE INTERRUPTS
64 64 01 D3 0407 1156 BITL #RB_CS_M_DRDY,RB_CS(R4) ;LOAD CSR (EXECUTION SUPPRESSED)
1C A4 01 CE 040A 1157 BNEQ 10$ ;IS DRIVE READY?
OC A4 01 CE 040C 1158 MNEGL #1,RB_CMD(R4) ;BRANCH IF SO
2E 11 0410 1159 10$: MNEGL #1,RB_DA(R4) ;INITIALIZE ENTIRE SUBSYSTEM
53 OE 9A 0414 1160 BRB SEEKI ;LOAD -1 IN DISK ADDRESS REGISTER
0093 C5 53 90 0416 1161 50$: MOVZBL #CDF_READHEAD,R3 ;INITIATE SEEK
00B9 31 0419 1162 MOVB R3,UCBSB_CEX(R5) ;SET FUNCTION AS READ HEADER
0421 1164 BRW XFER ;SAVE CASE INDEX
0421 1165 ;EXECUTE TRANSFER FUNCTION
```



```
0421 1167 .SBTTL POSITIONING FUNCTION EXECUTION
0421 1168 :
0421 1169 : POSITIONING FUNCTION EXECUTION
0421 1170 :
0421 1171 : FUNCTIONS INCLUDE:
0421 1172 :
0421 1173 : SEEK CYLINDER
0421 1174 :
0421 1175 : INPUTS:
0421 1176 : R3 - CASE INDEX
0421 1177 : R4 - DEVICE CSR ADDRESS
0421 1178 : R5 - UCB ADDRESS
0421 1179 :
0421 1180 : FUNCTIONAL DESCRIPTION:
0421 1181 :
0421 1182 : THE CYLINDER ADDRESS IS LOADED INTO THE DISK ADDRESS REGISTER.
0421 1183 : INTERRUPTS ARE LOCKED OUT, AND THE SEEK FUNCTION IS INITIATED.
0421 1184 : WITH INTERRUPT ENABLE. THE UNIT MUST BE SELECTED BEFORE LOADING
0421 1185 : THE CYLINDER ADDRESS (SO UCODE KNOWS WHETHER ITS AN RB80 OR RB02).
0421 1186 :
0421 1187 : WHEN THE FIRST INTERRUPT IS RECEIVED THE CHANNEL IS RELEASED (MUST
0421 1188 : OCCUR AT FORK LEVEL) AND THE COMPLETION INTERRUPT IS WAITED FOR.
0421 1189 : THE SEEK MAY COMPLETE WHILE AT FORK LEVEL SO A FLAG IS USED TO
0421 1190 : SYNCHRONIZE THE OPERATION.
0421 1191 :
0421 1192 :
0421 1193 :
0421 1194 POSIT: : POSITIONING FUNCTION
0421 1195 : GETUNIT : GET UNIT NUMBER IN R2
0429 1196 DSBINT UCB$B_DIPL(R5) : SAVE IPL AND LOCK OUT DEVICE INTERRUPTS
0430 1197 BISL3 FTAB[R3],R2,RB_CS(R4) : LOAD CSR (EXECUTION SUPPRESSED)
0437 1198 MOVL UCB$L_MEDIA(R5),RB_DA(R4) : LOAD CYLINDER ADDRESS IN DAR
043D 1199 MOVL UCB$L_MEDIA(R5),- : REMEMBER CURRENT DISK ADDRESS
0441 1200 UCB$L_DQ_CURDA(R5) : ...FOR SEEK OPTIMIZATION
0444 1201 :
0444 1202 SEEKI: : SEEK INITIATE
0444 1203 CKPWR 25$ : DISABLE INTERRUPTS, CHECK POWER
0451 1204 INITIATE : INITIATE THE FUNCTION
0458 1205 BISB #UCB$M_DQ_SIP,- : SIGNAL SEEK IN PROGRESS
045A 1206 UCB$B_DQ_FLAGS(R5) :
045D 1207 BDRVTYP RB02,T0$ : BRANCH IF RB02
0463 1208 :
0463 1209 : RB80'S INITIATE SEEKS VERY QUICKLY (APPROX 30 USECS). CONSEQUENTLY
0463 1210 : WE WAIT FOR THE SEEK TO INITIATE IN A LOOP, THEN CLEAR THE INITIATION
0463 1211 : INTERRUPT AND WAIT FOR THE COMPLETION INTERRUPT.
0463 1212 :
0463 1213 :
0463 1214 TIMEWAIT #3,#RB_CS_M_CRDY,- : WAIT FOR CONTROLLER READY
0463 1215 RB_CS(R4),C : ... 3*10 MICS
048A 1216 BLBC R0,10$ : BRANCH IF CONTROLLER STILL NOT READY
048D 1217 BICL #RB_CS_M_ATN,RB_CS(R4) : CLEAR INTERRUPT REQUEST FROM INITIATE
0494 1218 ENBINT : DROP IPL AND CLEANUP STACK
0497 1219 BRB 20$ :
0499 1220 :
0499 1221 : RB02'S CAN TAKE UPTO A FULL SECTOR TIME TO INITIATE AN INTERRUPT.
0499 1222 : CONSEQUENTLY WE TAKE TWO INTERUPTS, ONE FOR SEEK INITIATE, THE OTHER
0499 1223 : FOR SEEK COMPLETION
```

64 52 FC03 CF43 C9  
OC A4 00BC C5 D0  
00BC C5 D0  
00F2 C5  
01 88  
00C9 C5  
OC 50 E9  
000F0000 8F CA  
16 11

		0499	1224	:	WFIKPC	RETREG,#10	:WAIT FOR INITIATION INTERRUPT
		0499	1225	10\$:	BBS	#RB_CS_V_CE,-	:BRANCH IF SEEK INITIATE FAILED
28	00CC	0F	04A3	1226		UCB\$L_DQ_CS(R5),40\$	
	C5	E0	04A5	1227			
			04A9	1228	IOFORK		:DROP TO FORK IPL
			04AF	1229	RELCHAN		:RELEASE THE CHANNEL
			04B5	1230	DSBINT	UCB\$B_DIPL(R5)	:RETURN TO DEVICE IPL
05	00C9	00	04BC	1231	BBSC	#UCB\$V_DQ_SIP,-	:BRANCH IF SEEK NOT COMPLETED YET
	C5	E4	04BE	1232		UCB\$B_DQ_FLAGS(R5),30\$	
			04C2	1233	ENBINT		:RESTORE IPL
	10	11	04C5	1234	BRB	50\$	:DON'T WAIT FOR A SECOND INTERRUPT
			04C7	1235	WFIKPC	RETREG,#10	:WAIT FOR COMPLETION (CHANNEL RELEASED)
			04D1	1236	IOFORK		:DROP TO FORK IPL
01D6		31	04D7	1237	BRW	RETREG	:SEEK COMPLETION
			04DA	1238			



```
04DA 1240      .SBTTL  TRANSFER FUNCTION EXECUTION
04DA 1241
04DA 1242      :
04DA 1243      : TRANSFER FUNCTION EXECUTION
04DA 1244      :
04DA 1245      : FUNCTIONS INCLUDE:
04DA 1246      :
04DA 1247      : WRITE CHECK
04DA 1248      : WRITE DATA
04DA 1249      : READ DATA, AND
04DA 1250      : READ HEADER
04DA 1251      :
04DA 1252      : INPUTS:
04DA 1253      : R3      - CASE INDEX
04DA 1254      : R4      - DEVICE CSR ADDRESS
04DA 1255      : R5      - UCB ADDRESS
04DA 1256      :
04DA 1257      : FUNCTIONAL DESCRIPTION:
04DA 1258      :
04DA 1259      : THE TRANSFER PARAMETERS ARE LOADED INTO THE DEVICE REGISTERS, INTERRUPTS
04DA 1260      : ARE LOCKED OUT, THE FUNCTION IS INITIATED, AND A WAITFOR INTERRUPT AND
04DA 1261      : KEEP CHANNEL IS EXECUTED.
04DA 1262      :
04DA 1263      : UPON RETURN FROM THE INTERRUPT SERVICE ROUTINE, IF THE TRANSFER IS
04DA 1264      : COMPLETE, THE APPROPRIATE EXIT IS TAKEN. IF THE FUNCTION IS NOT COMPLETE
04DA 1265      : TRANSFER PARAMETERS ARE UPDATED AND A RETURN TO FDISPATCH IS EXECUTED TO
04DA 1266      : RE-ISSUE SEEK AND TRANSFER FUNCTIONS WHILE KEEPING CHANNEL AND UBA
04DA 1267      : RESOURCES. IF A DATA CHECK HAS BEEN REQUESTED, IT IS PERFORMED
04DA 1268      : BEFORE RETURNING TO FDISPATCH.
04DA 1269      :
04DA 1270      :
04DA 1271      : XFER:                                : TRANSFER FUNCTION EXECUTION
04DA 1272      :
04DA 1273      : LOAD UBA MAPS
04DA 1274      :
00C0 C5 B0 04DA 1275      : MOVW      UCBSW_BCR(R5),-      : GET BYTES LEFT TO TRANSFER AND -
7E A5 04DE 1276      : UCBSW_BCNT(R5)      : ASSUME ONLY ONE TRANSFER NEEDED
53 0E 91 04E0 1277      : CMPB      #CDF_READHEAD,R3      : IS THIS A READ HEADER OPERATION
32 13 04E3 1278      : BEQL      NOMAPS      : BRANCH IF SO, DON'T NEED MAPS
04E5 1279
04E5 1280
04E5 1281      :
04E5 1282      : COMPUTE SIZE OF THIS TRANSFER -- MAXIMUM = 1 TRACK
04E5 1283      :
52 44 A5 9A 04E5 1284      : MOVZBL UCBSB_SECTORS(R5),R2      : GET SECTORS/SURFACE
52 00BC C5 82 04E9 1285      : SUBB UCBSW_DA(R5),R2      : CALCULATE SECTORS LEFT ON SURFACE
52 0100 8F A4 04EE 1286      : MULW #256,R2      : COMPUTE BYTES REMAINING ON SURFACE
52 02 02 A4 04F3 1287      : BDRVTYP RB02,10$      : BRANCH IF AN RB02
52 09 E1 04F9 1288      : MULW #2,R2      : RB80 HAS 512 BYTE SECTORS
05 009A C5 04FC 1289      : BBC #10$V_SKIPSECTINH,-      : BRANCH NO SKIP SECTOR INHIBIT
52 0200 8F A0 04FE 1290      : UCBSW_FUNC(R5),10$      :
52 7E A5 B1 0502 1291      : ADDW #512,R2      : ALLOW ACCESS TO 32ND SECTOR
7E A5 52 B0 0507 1292 10$:      : CMPW UCBSW_BCNT(R5),R2      : ARE ADDITIONAL TRANSFERS REQUIRED?
0508 1293      : BLEQU 20$      : BRANCH IF NOT
050D 1294      : MOVW R2,UCBSW_BCNT(R5)      : STORE PARTIAL TRANSFER BYTE COUNT
0511 1295 20$:      : LOADUBAA      : LOAD UNIBUS MAP REGISTERS
0517 1296
```



```
0517 1297 :  
0517 1298 : MAPS LOADED (IF NECESSARY) AND BYTE COUNT DETERMINED.  
0517 1299 : LOAD BYTE COUNT  
0517 1300 :  
0517 1301 :  
0517 1302 : NOMAPS:  
0517 1303 : DSBINT UCB$B_DIPL(R5) :SAVE IPL AND LOCK OUT DEVICE INTERRUPTS  
52 7E A5 3C 051E 1304 : MOVZWL UCB$W_BCNT(R5),R2 :FETCH BYTE COUNT  
08 A4 52 CE 0522 1305 : MNEGL R2,RB_BC(R4) :SET NEGATIVE BYTE COUNT  
0526 1306 :  
0526 1307 :  
0526 1308 :  
0526 1309 : COMPUTE AND LOAD 18 BIT UNIBUS ADDRESS  
0526 1310 :  
0526 1311 :  
50 7C A5 3C 0526 1312 : MOVZWL UCB$W_BOFF(R5),R0 :FETCH BYTE OFFSET  
51 24 A5 D0 052A 1313 : MOVL UCB$L_CRB(R5),R1 :GET CRB ADDRESS  
34 A1 F0 052E 1314 : INSV CRB$L_INTD+VECSW_MAPREG(R1),- :INSERT STARTING MAP REGISTER  
50 09 09 0531 1315 : #9,#9,R0 :...NUMBER IN HIGH NINE BITS  
04 A4 50 D0 0534 1316 : MOVL R0,RB_BA(R4) :SET BUFFER ADDRESS  
0538 1317 :  
0538 1318 :  
0538 1319 :  
0538 1320 : PERFORM R80 TRACK-TO-TRACK SPIRALLING  
0538 1321 : THE R80 CAN CHANGE HEADS JUST BY LOADING A SEEK COMMAND,  
0538 1322 : AND LOADING THE DAR. WE TAKE ADVANTAGE OF THIS FEATURE  
0538 1323 : TO REDUCE SEEK TIMES  
0538 1324 :  
0538 1325 :  
0538 1326 :  
00BD C5 91 0540 1327 : GETUNIT :GET UNIT NUMBER IN R2  
00F3 C5 0544 1328 : CMPB UCB$L_MEDIA+1(R5),- :COMPARE DESIRED TRACK  
18 13 0547 1329 : UCB$L_DQ_CURDA+1(R5) :...TO CURRENT TRACK  
53 0E 91 0549 1330 : BEQL 20$ :BRANCH IF ON TRACK  
13 13 054C 1331 : CMPB #CDF_READHEAD,R3 :IS THIS A READ HEADER OPERATION?  
64 52 FAEE CF C9 054E 1332 : BEQL 20$ :BRANCH IF SO, DON'T CHANGE HEADS  
0C A4 00BC C5 D0 0554 1333 : BISL3 FTAB<CDF_SEEK*4>,R2,- :SET CONTROLLER TO SEEK MODE  
00BD C5 90 0554 1334 : RB_CS(R4) :...  
00F3 C5 055A 1335 : MOVL UCB$L_MEDIA(R5),RB_DA(R4) :DO A HEAD SELECT  
055E 1336 : MOV B UCB$L_MEDIA+1(R5),- :UPDATE CURRENT DISK ADDRESS  
0561 1337 : UCB$L_DQ_CURDA+1(R5) :...WITH NEW TRACK  
0561 1338 :  
0561 1339 : EXECUTE THE TRANSFER FUNCTION --  
0561 1340 : NOTE: THE FUNCTION MUST BE SPECIFIED BEFORE LOADING THE DAR  
0561 1341 : BECAUSE THE UCODE MUST KNOW WHETHER THE TRANSFER IS TO AN  
0561 1342 : RB02 OR AN RB80.  
0561 1343 :  
0561 1344 :  
52 07 009A C5 E1 0561 1345 20$: BBC #IOSV_SKPSECTINH,- :BRANCH NO SKIP SECTOR INHIBIT  
08400000 8F C8 0563 1346 : UCB$W_FUNC(R5),30$ :...  
64 52 FAC5 CF43 C9 0567 1347 : BISL #RB_CS_M_SSE1- :INHIBIT SKIP SECTOR ERRORS  
0C A4 00BC C5 D0 056E 1348 : !RB-CS-M-ASSI,R2 :...AND AUTOMATIC SKIP SECTORING  
0575 1349 30$: BISL3 FTAB[R3],R2,RB_CS(R4) :LOAD UNIT NUMBER AND FUNCTION  
0582 1350 : CKPWR BRW RETREG :DISABLE INTERRUPTS, CHECK POWER  
0588 1351 : MOVL UCB$L_MEDIA(R5),RB_DA(R4) :SET DESIRED DISK ADDRESS  
058F 1352 : INITIATE :INITIATE THE FUNCTION  
WFIKPC RETREG,#10 :WAITFOR INTERRUPT AND KEEP CHANNEL
```



```
0599 1354 :  
0599 1355 :THE R80 PRODUCES SPURIOUS ATTENTION BITS ON XFER'S. UNTIL FIXED WE  
0599 1356 :FOLLOW EACH TRANSFER WITH AN EXPLICIT CLEAR OF THE UNIT'S ATTENTION BIT.  
0599 1357 :  
51 0F 10 78 0599 1358 ASHL #RB_CS_V_ATN,#^XOF,R1 ;FORM BIT MASK  
50 50 54 A5 3C 059D 1359 MOVZWL UCBSW_UNIT(R5),R0 ;FETCH UNIT NUMBER  
00 50 10 C0 05A1 1360 ADDL #RB_CS_V_ATN,R0 ;POINT INTO MASK  
00 51 50 E5 05A4 1361 BBCC R0,R1,50$ ;CLEAR THIS UNIT'S ATTENTION BIT  
64 51 CA 05A8 1362 50$: BICL R1,RB_CS(R4) ;CLEAR THIS UNIT'S BIT IN THE CSR  
05AB 1363 IOFORK ;CREATE FORK PROCESS (RETURN TO ISR)  
03 00CC C5 E1 05B1 1364 BBC #RB_CS_V_CE,- ;BRANCH IF NO ERRORS  
00F6 31 05B3 1365 UCBSL_DQ_CS(R5),RETHDR ;  
05B7 1366 BRW_RETREG: ; WORD DISPLACEMENT, UNCONDITIONAL BRANCH  
05B7 1367 BRW RETREG ;RETURN REGISTERS  
05BA 1368
```

```
05BA 1370      .SBTTL  TRANSFER POST PROCESSING
05BA 1371
05BA 1372 :
05BA 1373 : PURGE DATAPATH -- NOTE: THE DATAPATH IS NOT PURGED BECAUSE THIS
05BA 1374 : DRIVER IS SPECIFIC TO THE VAX730 PROCESS WHICH DOES NOT REQUIRE
05BA 1375 : DATAPATH PURGING. CONSEQUENTLY THE DATAPATH REGISTER WILL ALWAYS
05BA 1376 : BE ZERO IN ERRLOG AND DIAGNOSTIC BUFFERS.
05BA 1377 :
05BA 1378
05BA 1379
05BA 1380 :
05BA 1381 : RETURN HEADER INFORMATION FOR READ HEADER FUNCTION --
05BA 1382 : IF AN INTERNAL READY HEADER THEN SIMPLY EXIT.
05BA 1383 :
05BA 1384
05BA 1385 RETHDR:      ;RETURN HEADER INFO
05BA 1386      CMPB      #CDF READHEAD, -      ;WAS THIS A READ HEADER?
05BF 1387      UCB$B_CEX(R5)      ;...
05BF 1388      BNEQ     WRITECHK      ;BRANCH IF NOT
05C1 1389      CMPB      #IOS READHEAD, -      ;INTERNAL READ HEADER?
05C6 1390      UCB$B_FEX(R5)      ;...
05C6 1391      BNEQ     BRW_RETREG      ;BRANCH IF SO
05C8 1392      PUSHL    UCB$B_SVAPTE(R5)      ;SAVE ADDRESS OF PTE
05CB 1393      MOVAB    UCB$W_DQ_HDR1(R5),R1      ;SET ADDRESS OF INTERNAL BUFFER
05D0 1394      MOVL     #6,R2      ;SET NUMBER OF BYTES TO MOVE
05D3 1395      CMPW     R2,UCB$W_BCNT(R5)      ;ROOM FOR FULL HEADER?
05D7 1396      BLEQU    30$      ;BRANCH IF SO
05D9 1397      MOVZWL   UCB$W_BCNT(R5),R2      ;SET LENGTH OF PARTIAL HEADER
05DD 1398      SUBW2    R2,UCB$W_BCR(R5)      ;UPDATE BYTE COUNT REMAINING
05E2 1399      JSB      G^IOCSMOVTOUSER      ;MOVE HEADER TO USER BUFFER
05E8 1400      POPL     UCB$B_SVAPTE(R5)      ;RESTORE ADDRESS OF PTE
05EC 1401      BRB      BRW_RETREG      ;TERMINATE FUNCTION
05EE 1402
```

0093 C5 0E 91 05BA 1385 RETHDR: ;RETURN HEADER INFO  
0092 C5 0E 91 05BA 1386 CMPB #CDF READHEAD, - ;WAS THIS A READ HEADER?  
51 00EC C5 9E 05BF 1387 UCB\$B\_CEX(R5) ;...  
52 06 D0 05BF 1388 BNEQ WRITECHK ;BRANCH IF NOT  
7E A5 52 B1 05C1 1389 CMPB #IOS READHEAD, - ;INTERNAL READ HEADER?  
52 04 1B 05C6 1390 UCB\$B\_FEX(R5) ;...  
00C0 C5 52 A2 05C6 1391 BNEQ BRW\_RETREG ;BRANCH IF SO  
00000000 GF 16 05C8 1392 PUSHL UCB\$B\_SVAPTE(R5) ;SAVE ADDRESS OF PTE  
78 A5 8ED0 05CB 1393 MOVAB UCB\$W\_DQ\_HDR1(R5),R1 ;SET ADDRESS OF INTERNAL BUFFER  
C9 11 05D0 1394 MOVL #6,R2 ;SET NUMBER OF BYTES TO MOVE  
05D3 1395 CMPW R2,UCB\$W\_BCNT(R5) ;ROOM FOR FULL HEADER?  
05D7 1396 BLEQU 30\$ ;BRANCH IF SO  
05D9 1397 MOVZWL UCB\$W\_BCNT(R5),R2 ;SET LENGTH OF PARTIAL HEADER  
05DD 1398 SUBW2 R2,UCB\$W\_BCR(R5) ;UPDATE BYTE COUNT REMAINING  
05E2 1399 JSB G^IOCSMOVTOUSER ;MOVE HEADER TO USER BUFFER  
05E8 1400 POPL UCB\$B\_SVAPTE(R5) ;RESTORE ADDRESS OF PTE  
05EC 1401 BRB BRW\_RETREG ;TERMINATE FUNCTION  
05EE 1402



```
05EE 1404 .SBTTL DATA CHECK AND PARAMETER UPDATE
05EE 1405
05EE 1406
05EE 1407 : PERFORM WRITE CHECK, IF REQUESTED
05EE 1408
05EE 1409
05EE 1410 WRITECHK: :WRITECHECK AFTER PARTIAL TRANSFER
11 009A 0E E1 05EE 1411 BBC #IOSV_DATACHECK,- :IF CLR - DATA CHECK NOT REQUESTED
01 05F0 1412 UCBSW_FUNC(R5),UPDATE
00C9 01 E4 05F4 1413 BBSC #UCBSV_DQ_DIP,- :CLEAR DATA CHECK IN PROGRESS
0B 05F6 1414 UCBSB_DQ_FLAGS(R5),- :...AND BRANCH IF SET
02 05F9 1415 UPDATE :SET DATA CHECK IN PROGRESS
00C9 02 88 05FA 1416 BISB #UCBSM_DQ_DIP,- :SET DATA CHECK IN PROGRESS
53 0A 9A 05FC 1417 UCBSB_DQ_FLAGS(R5) :SET CASE INDEX TO WRITE CHECK
FF12 31 05FF 1418 MOVZBL #IOS_WRITECHECK,R3 :BRANCH TO PERFORM WRITE CHECK
0602 1419 BRW NOMAPS
0605 1420
0605 1421
0605 1422 :
0605 1423 : UPDATE BUFFER ADDRESS, CURRENT DISK ADDRESS, AND BYTES REMAINING
0605 1424 : FOR NEXT TRANSFER
0605 1425 :
0605 1426
0605 1427 UPDATE: :UPDATE TRANSFER PARAMETERS
50 7E A5 3C 0605 1428 MOVZWL UCBSW_BCNT(R5),R0 :FETCH BYTES TRANSFERRED
00C0 C5 50 A2 0609 1429 SUBW R0,UCBSW_BCR(R5) :UPDATE BYTES REMAINING TO XFER
A7 13 060E 1430 BEQL BRW_RETREG :BRANCH IF TRANSFER COMPLETE
0610 1431
0610 1432 :
0610 1433 : COMPUTE NUMBER OF 512 BYTE BLOCKS TRANSFERED
0610 1434
50 50 F9 8F 78 0610 1435 10$: ASHL #-7,R0,R0 :COMPUTE PAGES * 4
78 A5 50 C0 0615 1436 ADDL R0,UCBSL_SVAPTE(R5) :UPDATE THE ADDRESS OF THE PTE
78 A5 03 8A 0619 1437 BICB2 #^X3,UCBSL_SVAPTE(R5) :ROUND DOWN TO FULL PAGES (RL02'S!)
50 50 FF 8F 78 061D 1438 ASHL #-1,R0,R0 :CONVERT TO 256 BYTE SECTORS
0622 1439 BDRVTYP RB80,15$ :BRANCH IF RB80
OF 50 E9 0628 1440 BLBC R0,20$ :CHECK FOR ODD SECTOR ADDRESSING
7D A5 7D A5 96 062B 1441 INCB UCBSW_BOFF+1(R5) :ADD ^X100 TO BOFF
FE 8F 8A 062E 1442 BICB #^XFE,UCBSW_BOFF+1(R5) :MAKE BOFF MODULO ^X200
05 11 0633 1443 BRB 20$ :CONTINUE IN COMMON
0635 1444
50 50 FF 8F 78 0635 1445 15$: ASHL #-1,R0,R0 :CONVERT TO 512 BYTE SECTORS
00BC C5 50 80 063A 1446 20$: ADDB R0,UCBSW_DA(R5) :UPDATE SECTOR
00BC C5 91 063F 1447 CMPB UCBSW_DAT(R5),- :COMPARE UPDATED SECTOR
44 A5 0643 1448 UCBSB_SECTORS(R5) :...TO SECTORS PER TRACK
28 1F 0645 1449 BLSSU 50$ :BRANCH IF MORE REMAIN
OE 12 0647 1450 BNEQ 30$ :BRANCH IF PAST LOGICAL END OF TRACK
0649 1451 BDRVTYP RB02,30$ :BRANCH IF RB02
02 009A 09 E1 064F 1452 BBC #IOSV_SKPSECTINH,- :BRANCH NO SKIP SECTOR INHIBIT
C5 0651 1453 UCBSW_FUNC(R5),30$ :...
0655 1454 :
0655 1455 : THIS IS AN R80 DRIVE, ON THE LAST LOGICAL SECTOR, AND SKIP SECTOR
0655 1456 : INHIBIT IS SET -- THERE IS ONE PHYSICALLY ACCESSABLE BLOCK REMAINING, SO
0655 1457 : CONTINUE ON THE SAME TRACK
0655 1458 :
18 11 0655 1459 BRB 50$ :ONE MORE SECTOR REMAINS
0657 1460
```



DQDRIVER  
V04-000

L 14  
- VAX/VMS RB730:RB02/RB80 DISK DRIVER  
DATA CHECK AND PARAMETER UPDATE

15-SEP-1984 23:49:22 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:46 [DRIVER.SRC]DQDRIVER.MAR;1

Page 33  
(1)

00BC	C5	94	0657	1461	30\$:	CLRB	UCBSW_DA(R5)	:CLEAR SECTOR ADDRESS
00BD	C5	96	065B	1462		INCB	UCBSW_DA+1(R5)	:INCREMENT TRACK
00BD	C5	91	065F	1463		CMPB	UCBSW_DA+1(R5) -	:COMPARE UPDATED TRACK
45	A5		0663	1464			UCBSB_TRACKS(R5)	:...TO TRACKS PER CYLINDER
	08	1F	0665	1465		BLSSU	50\$	:BRANCH IF MORE REMAIN
00BD	C5	94	0667	1466		CLRB	UCBSW_DA+1(R5)	:RESET DESIRED TRACK (SURFACE) TO 0
00BE	C5	B6	066B	1467		INCW	UCBSW_DC(R5)	:INCREMENT CYLINDER
			066F	1468				
FB18		31	066F	1469	50\$:	BRW	FDISPATCH	:MORE BYTES REMAINING - CONTINUE
			0672	1470				



```
0672 1472 .SBTTL SPECIAL CONDITION (POWER, TIMEOUT)
0672 1473
0672 1474 :
0672 1475 : SPECIAL CONDITION EXIT (POWER FAILURE OR DEVICE TIMEOUT)
0672 1476 :
0672 1477
0672 1478 SPECOND:
0672 1479 MNEGL #1,UCBSL DQ CURDA(R5) ;DISABLE SEEK OPTIMIZATION
0677 1480 BBSC #UCBSV POWER,- ;IF SET - POWER FAILURE
0679 1481 UCBSW_STS(R5),PWRFAIL ;...ELSE TIMEOUT
067C 1482 SETIPL UCBSB-FIPL(R5) ;TIMEOUTS ENTER AT DEVICE IPL
0680 1483 JSB G*ERL$DEVICTMO ;LOG DEVICE TIMEOUT
0686 1484 CLRW UCBSW BCNT(R5) ;SET ZERO BYTES TRANSFERED
0689 1485 BICW #UCBSM TIMEOUT,UCBSW_STS(R5) ;CLEAR TIMEOUT STATUS
068F 1486 MOVZWL #SS$ TIMEOUT,R0 ;SET DEVICE TIMEOUT STATUS
0694 1487 DECB UCBSB_ERTCNT(R5) ;ANY ERROR RETRIES REMAINING?
0698 1488 BLEQ 10$ ;BRANCH IF NOT
069A 1489 BRW RESETDRIVE ;RETRY THE FUNCTION
069D 1490 10$: BRW FUNCXT ;GIVE UP
06A0 1491
06A0 1492 PWRFAIL:
06A0 1493 BICW #UCBSM POWER,UCBSW_STS(R5) ;POWER FAILURE
06A4 1494 MOVL UCBSL_IRP(R5),R3 ;CLEAR POWER FAILURE BIT
06A8 1495 MOVQ IRP$S_SVAPTE(R3),- ;GET ADDRESS OF I/O PACKET
06AB 1496 UCBSL_SVAPTE(R5) ;RESTORE TRANSFER PARAMETERS
06AD 1497 BRW PREPROCESS ;RETURN TO PREPROCESS UCB FIELDS
06B0 1498
```

00F2 C5 01 CE  
05 E4  
24 64 A5  
00000000 GF 16  
7E A5 B4  
64 A5 0040 8F AA  
50 022C 8F 3C  
0080 C5 97  
03 15  
FC1B 31  
FC81 31  
64 A5 20 AA  
53 58 A5 D0  
2C A3 7D  
78 A5 06AB  
FA6B 31  
06AD  
06B0



```
.SBTTL  HARDWARE FUNCTION EXIT PROCESSING

06B0 1500
06B0 1501
06B0 1502
06B0 1503 : DETERMINE EXIT - SPECIAL CONDITION, FATAL ERROR, RETRIABLE ERROR, OR SUCCESS
06B0 1504 :
06B0 1505
B3 06B0 1506 RETREG: BITW #UCBSM_POWER!- :POWER FAIL
06B1 1507 UCBSM_TIMEOUT- :...OR DEVICE TIMEOUT
06B1 1508 UCBSW_STS(R5) :...IN STATUS WORD?
06B6 1509 BNEQ SPECORD :BRANCH IF SO -- SPECIAL CONDITION
51 00CC C5 D0 06B8 1510 MOVL UCBSL_DQ_CS(R5),R1 :FETCH CSR
7B 51 9F E1 06BD 1511 BBC #RB_CS_V_CE,R1,SUCCESS :BRANCH IF NO ERRORS
00F2 C5 01 CE 06C1 1512 MNEGL #1,UCBSL_DQ_CURDA(R5) :DISABLE SEEK OPTIMIZATION
00D4 C5 A0 06C6 1513 ADDW UCBSL_DQ_BC(R5),- :ADD NEGATIVE BYTE COUNT REMAINING
7E A5 52 00DC C5 D0 06CA 1514 UCBSW_BCNT(R5) :...TO PARTIAL TRANSFER COUNT
06CC 1515 MOVL UCBSL_DQ_MP(R5),R2 :FETCH MPR
06D1 1516
06D1 1517 :
06D1 1518 : CHECK TO SEE IF THERE IS ANY ERROR OTHER THAN OPI. IF NO OTHER
06D1 1519 : ERRORS, AND THE DRIVE IS NOT READY, THEN ASSUME IT WAS SIMPLY SPUN DOWN
06D1 1520 :
51 00B07801 8F D3 06D1 1521 BITL #RB_CS_M_SSE - :SKIP SECTOR ERROR
06D8 1522 !RB_CS_M_DE - :... OR DRIVE ERROR
06D8 1523 !RB_CS_M_NXM - :... OR NON-EXISTANT MEMORY
06D8 1524 !RB_CS_M_DLT - :... OR DATA LATE
06D8 1525 !RB_CS_M_DCK - :... OR DATA CHECK
06D8 1526 !RB_CS_M_DRDY,R1 :... OR DRIVE READY?
06D8 1527 BNEQ 20$ :BRANCH IF SO
00 64 A5 E5 06DA 1528 10$: BBCC #UCBSV_VALID,- :CLEAR VALID BIT
50 01A4 8F 3C 06DC 1529 UCBSW_STS(R5),15$ :
FC3A 31 06DF 1530 15$: MOVZWL #SS$_MEDOFL,R0 :SET MEDIUM OFFLINE STATUS
06E4 1531 BRW FUNCXT :RETURN
06E7 1532
00000000*GF 16 06E7 1533 20$: JSB G*ERL$DEVICERR :ALLOCATE AND FILL ERROR MESSAGE BUFFER
53 51 0D E0 06ED 1534 BBS #RB_CS_V_NXM,R1,FATAL :BRANCH IF NON-EXISTENT MEMORY
4F 51 17 E0 06F1 1535 BBS #RB_CS_V_SSE,R1,FATAL :BRANCH IF SKIP SECTOR ERROR
2F 51 0E E1 06F5 1536 BBC #RB_CS_V_DE,R1,RETRY :BRANCH IF NO DRIVE ERRORS
06F9 1537 BDRV Typ RB02,50$ :BRANCH IF RB02
06FF 1538
06FF 1539 :
06FF 1540 :CLASSIFY RB80 ERRORS AS FATAL OR RETRIABLE
06FF 1541 :
06FF 1542 :
D7 52 09 E1 06FF 1543 BBC #RB_MP_V_PLGV,R2,10$ :BRANCH IF PLUG NOT VALID
23 11 0703 1544 BRB RETRY :
0705 1545
0705 1546 :
0705 1547 :CLASSIFY RB02 ERRORS AS FATAL OR RETRIABLE
0705 1548 :
1D 52 06 00 ED 0705 1549 50$: CMPZV #0,#6,R2,- :STATE OK? ...COVER CLOSED
070A 1550 #RB_MP_M_HO- :...HEADS OUT
070A 1551 !RB_MP_M_BH- :...BRUSHES HOME
070A 1552 !RB_MP_C_SLM :...READY TO GO
CE 12 070A 1553 BNEQ 10$ :BRANCH IF NOT
070C 1554
07 52 09 E1 070C 1555 BBC #RB_MP_V_VC,R2,60$ :BRANCH IF VOLUME VALID
0B E5 0710 1556 BBCC #UCBSV_VALID,- :CLEAR VALID BIT
```



```
00 64 A5      0712 1557      UCBSW_STS(R5),55$      ;...
2D      11 0715 1558 55$:   BRB      FATAL      ;RETURN
      0717 1559
04 52 0D      E1 0717 1560 60$:   BBC      #RB_MP-V-WL,R2,70$      ;BRANCH IF NOT WRITE LOCKED
25 52 0A      E0 071B 1561      BBS      #RB_MP-V-WGE,R2,FATAL      ;IF WL & WGE THEN WL ERROR
52 0000CD00 BF D3 071F 1562 70$:   BITL     #RB_MP-M-WDE-      ;WRITE DATA ERROR
      0726 1563      ;RB_MP-M-HCE-      ;...OR HEAD CURRENT ERROR
      0726 1564      ;RB_MP-M-SPD-      ;...OR SPINDLE SPEED ERROR
      0726 1565      ;RB_MP-M-WGE-      ;...OR WRITE GATE ERROR
      0726 1566      ;RB_MP-M-DSE,R2      ;...OR DRIVE SELECT ERROR?
1C      12 0726 1567      BNEQ     FATAL      ;BRANCH IF SO
      0728 1568
      0728 1569      ;
      0728 1570      ; RETRIABLE ERROR EXIT
      0728 1571      ;
      0728 1572
      0728 1573 RETRY:   BBS      #IOSV_INHRETRY,-      ;BRANCH IF RETRIES INHIBITED
16 009A C5      E0 072A 1574      UCBSW_FUNC(R5),FATAL      ;...
50 009C D5      98 072E 1575      CVTBL     @UCBSL_DPC(R5),R0      ;GET BRANCH DISPLACEMENT
50 009C C5      C0 0733 1576      ADDL      UCBSL_DPC(R5),R0      ;COMPUTE JUMP ADDRESS -1
      50 D6 0738 1577      INCL      R0      ;COMPUTE JUMP ADDRESS
      60 17 073A 1578      JMP        (R0)      ;RETURN TO ERROR ROUTINE
      073C 1579
      073C 1580      ;
      073C 1581      ; SUCCESSFUL OPERATION EXIT
      073C 1582      ;
      073C 1583
009C C5 D6      073C 1584 SUCCESS:INCL UCBSL_DPC(R5)      ;ADJUST TO CORRECT RETURN ADDRESS
009C D5 17      0740 1585      JMP        @UCBSL_DPC(R5)      ;RETURN TO DRIVER
      0744 1586
      0744 1587      ;
      0744 1588      ; FATAL ERROR EXIT
      0744 1589      ;
      0744 1590
FB7B 31 0744 1591 FATAL:   BRW      FATALERR      ;FATAL ERROR EXIT
      0747 1592
```



```
0747 1594 .SBTTL INTERRUPT SERVICE ROUTINE
0747 1595
0747 1596 :++
0747 1597 :DQ$INT - RB730 INTERRUPT SERVICE ROUTINE
0747 1598
0747 1599 :FUNCTIONAL DESCRIPTION:
0747 1600 :
0747 1601 :THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT
0747 1602 :OCCURS ON AN RB730 DISK CONTROLLER. IF THE INTERRUPT IS NOT EXPECTED,
0747 1603 :THE UNSOLICITED INTERRUPT ROUTINE DISMISSES THE INTERRUPT. IF
0747 1604 :THE INTERRUPT IS EXPECTED, DEVICE REGISTERS ARE SAVED AND THE
0747 1605 :INTERRUPTING UNIT IS DETERMINED. THE DRIVER IS CALLED AT ITS INTERRUPT
0747 1606 :RETURN ADDRESS. THE DRIVER FORKS, CAUSING A RETURN TO THIS ROUTINE
0747 1607 :WHICH CONTINUES TO SCAN THE ATTENTION SUMMARY REGISTER IN CASE
0747 1608 :ANY MORE DRIVES REQUIRE SERVICE. AFTER THE LAST DRIVE IS SERVICED
0747 1609 :THIS ROUTINE RESTORES GENERAL REGISTERS AND DISMISSES THE INTERRUPT.
0747 1610
0747 1611 :INPUTS:
0747 1612 :
0747 1613 :00(SP) - POINTER TO ADDRESS OF THE IDB
0747 1614 :04(SP) - SAVED R0
0747 1615 :08(SP) - SAVED R1
0747 1616 :12(SP) - SAVED R2
0747 1617 :16(SP) - SAVED R3
0747 1618 :20(SP) - SAVED R4
0747 1619 :24(SP) - SAVED R5
0747 1620 :28(SP) - PC AT THE TIME OF THE INTERRUPT
0747 1621 :32(SP) - PSL AT THE TIME OF THE INTERRUPT
0747 1622
0747 1623 :OUTPUTS:
0747 1624 :
0747 1625 :DEVICE REGISTERS ARE SAVED, IPL IS LOWERED TO FORK LEVEL, THE
0747 1626 :INTERRUPT IS DISMISSED, ALL REGISTERS EXCEPT R0-R5 ARE PRESERVED.
0747 1627 :
0747 1628 :--
0747 1629
0747 1630 DQ_REI: ;INTERRUPT EXIT CODE
0747 1631 ADDL #4,SP ;POP IDB ADDRESS
074A 1632 POPR #*M<R0,R1,R2,R3,R4,R5> ;RESTORE R0-R5
074C 1633 REI ;RETURN FROM INTERRUPT
074D 1634
074D 1635 DQ_INT: ;INTERRUPT SERVICE ROUTINE
074D 1636 MOVL @ (SP),R3 ;FETCH ADDRESS OF IDB
0751 1637 MOVL IDB$CSR(R3),R4 ;GET ADDRESS OF CSR
0754 1638 MOVL IDB$OWNER(R3),R5 ;GET OWNER UCB ADDRESS
0758 1639 BEQL 12$ ;BRANCH IF NOT OWNED
075A 1640 BBSC #UCB$V_INT,- ;BRANCH IF INTERRUPT EXPECTED
075C 1641 UCB$W_STS(R5),40$ ;...
075F 1642 :
075F 1643 :SCAN ATTENTION BITS TO DETERMINE INTERRUPTING DRIVE
075F 1644 :
075F 1645 10$: MOVL @ (SP),R3 ;FETCH ADDRESS OF IDB
0763 1646 MOVL IDB$CSR(R3),R4 ;GET ADDRESS OF CSR
0766 1647 12$: MOVL RB_CSTR4),R1 ;GET CSR
0769 1648 FFS #RB_CS_V_ATN,- ;FIND REQUESTING DRIVE
076B 1649 #RB_CS_S_ATN,R1,R1 ;...
076E 1650 BEQL DQ_REI ;BRANCH IF NO MORE DRIVES TO SERVICE
```



```
50 0F 10 78 0770 1651 ASHL #RB_CS_V_ATN,#^XOF,R0 ;PREPARE MASK OF ATTENTION BITS
00 50 51 E5 0774 1652 BBCC R1,R0,T5$ ;CLEAR THIS UNIT'S BIT IN THE MASK
64 64 50 CA 0778 1653 15$: BICL R0,RB_CS(R4) ;CLEAR THIS UNIT'S BIT IN THE CSR
51 10 C2 077B 1654 SUBL #RB_CS_V_ATN,R1 ;COMPUTE UNIT NUMBER
52 51 08 78 077E 1655 ASHL #RB_CS_V_DS,R1,R2 ;MOVE UNIT INTO DRIVE SELECT BITS
52 52 C9 0782 1656 BISL3 R2 ;SELECT THE UNIT
0784 1657 #RB_CS_M_IE- ;...WITH INTERRUPT ENABLE
0784 1658 !RB_CS_M_CRDY,- ;...AND CONTROLLER READY
0784 1659 RB_CS(R4) ;...DEVICE CSR
64 000000C0 8F D0 078A 1660 MOVL IDB$L_UCBLST(R3)[R1],R5 ;GET ADDRESS OF UCB
55 18 A341 13 078F 1661 BEQL 25$ ;BRANCH IF UCB WAS NOT FOUND
01 E4 0791 1662 BBSC #UCB$V_INT,- ;BRANCH IF INTERRUPT WAS EXPECTED
10 64 A5 0793 1663 UCB$W_STS(R5),40$ ;...
00CC C5 64 D0 0796 1664 MOVL RB_CSTR4),UCB$L_DQ_CS(R5) ;SAVE CSR
00 00 E4 079B 1665 BBSC #UCB$V_DQ_SIP,- ;BRANCH IF SEEK IN PROGRESS
C5 00C9 C5 079D 1666 UCB$B_DQ_FLAGS(R5),12$ ;...AND CONTINUE SCANNING
00C8 30 07A1 1667 25$: BSBW DQ_UNEXINT ;HANDLE UNEXPECTED INTERRUPT
C0 11 07A4 1668 BRB 12$ ;CONTINUE SCANNING
07A6 1670
07A6 1671 ;
07A6 1672 ;HERE WHEN UNIT DETERMINED, INTERRUPT EXPECTED, DRIVE SELECTED
07A6 1673 ;AND STATUS AVAILABLE
07A6 1674 ;
0093 C5 0E 91 07A6 1675 40$: CMPB #CDF_READHEAD,UCB$B_CEX(R5) ;READ HEADER FUNCTION?
12 12 07AB 1676 BNEQ 50$ ;IF NEQ - NO
00EC C5 10 A4 F7 07AD 1677 CVTLW RB_MP(R4),UCB$W_DQ_HDR1(R5) ;SAVE SECTOR HEADER INFORMATION
00EE C5 10 A4 F7 07B3 1678 CVTLW RB_MP(R4),UCB$W_DQ_HDR2(R5) ;... (THIS MUST BE DONE EVEN
00F0 C5 10 A4 F7 07B9 1679 CVTLW RB_MP(R4),UCB$W_DQ_HDR3(R5) ;...FOR INTERNAL READ HEADERS)
07BF 1680
64 00008000 8F D3 07BF 1681 50$: BITL #RB_CS_M_CE,RB_CS(R4) ;COMPOSITE ERROR?
14 12 07C6 1682 BNEQ 80$ ;BRANCH IF SO
01 E0 07C8 1683 BBS #UCB$V_DIAGBUF,- ;BRANCH IF DIAGNOSTIC BUFFER
0F 68 A5 07CA 1684 UCB$W_DEVSTS(R5),80$ ;...IS PRESENT
00CC C5 64 D0 07CD 1685 MOVL RB_CSTR4),UCB$L_DQ_CS(R5) ;SAVE CSR ONLY
07D2 1686
07D2 1687 ;
07D2 1688 ;RETURN TO FUNCTION EXECUTION
07D2 1689 ;
53 10 A5 7D 07D2 1690 60$: MOVQ UCB$L_FR3(R5),R3 ;RESTORE DRIVER CONTEXT
OC B5 16 07D6 1691 JSB @UCB$L_FPC(R5) ;CALL DRIVER AT INTERRUPT RETURN ADDRESS
FF83 31 07D9 1692 BRW 10$ ;CHECK FOR MORE DRIVES TO SERVICE
07DC 1693
07DC 1694 ;
07DC 1695 ;DEVICE ERROR OR DIAGNOSTIC BUFFER -- SAVE THE DEVICE REGISTERS
07DC 1696 ;AND RESET THE DRIVE
07DC 1697 ;
02 10 07DC 1698 80$: BSBB DQ_REGSAVE ;SAVE DEVICE REGISTERS
F2 11 07DE 1699 BRB 60$ ;CONTINUE
07E0 1700
```



```
07E0 1702 .SBTTL REGISTER SAVE ROUTINE
07E0 1703 :++
07E0 1704 :
07E0 1705 DQ_REGSAVE - REGISTER SAVE ROUTINE
07E0 1706 :
07E0 1707 FUNCTIONAL DESCRIPTION:
07E0 1708 :
07E0 1709 THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
07E0 1710 REGISTERS IN THE UCB. STATUS IS OBTAINED FROM THE DRIVE AND IF AN
07E0 1711 ERROR HAS OCCURED THEN THE DRIVE IS RESET (COULD BE HERE BECAUSE
07E0 1712 DIAGNOSTIC BUFFER PRESENT.
07E0 1713 :
07E0 1714 INPUTS:
07E0 1715 :
07E0 1716 R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
07E0 1717 R5 - ADDRESS OF UNIT CONTROL BLOCK (UCB)
07E0 1718 :
07E0 1719 OUTPUTS:
07E0 1720 :
07E0 1721 R0-R3 - DESTROYED
07E0 1722 THE DEVICE REGISTERS ARE SAVED IN THE UCB.
07E0 1723 :
07E0 1724 :--
07E0 1725 :
07E0 1726 DQ_REGSAVE:
07E0 1727 MOVAB RB CS(R4),R2 ;REGISTER SAVE ROUTINE
07E0 1728 MOVAB UCB$$_DQ_CS(R5),R3 ;GET ADDRESS OF CONTROL STATUS REGISTER
07E0 1729 MOVL (R2)+,(R3)+ ;GET ADDRESS OF REGISTER SAVE AREA
07E0 1730 MOVL (R2)+,(R3)+ ;SAVE CONTROL STATUS REGISTER
07E0 1731 MOVL (R2)+,(R3)+ ;SAVE BUFFER ADDRESS REGISTER
07E0 1732 MOVL (R2)+,(R3)+ ;SAVE BYTE COUNT REGISTER
07E0 1733 MOVL (R2)+,(R3)+ ;SAVE DISK ADDRESS REGISTER
07E0 1734 CVTLW (R2)+,UCB$$_EC1(R5) ;SAVE MPR REGISTER
07E0 1735 CVTLW (R2)+,UCB$$_EC2(R5) ;SAVE ECC POSITION REGISTER
07E0 1736 :
07E0 1737 : GET STATUS
07E0 1738 :
07E0 1739 GETUNIT ;GET UNIT NUMBER IN R2
07E0 1740 CMPZV #RB_CS_V_FCODE,- ;WAS ORIGIANL FUNCTION A GET STATUS?
07E0 1741 #RB_CS_S_FCODE,-
07E0 1742 UCB$$_DQ_CS(R5),-
07E0 1743 #F GETSTATUS
07E0 1744 BEQL 20$ ;BRANCH IF SO (USE ORIGINAL STATUS)
07E0 1745 MOVZBL #-1,UCB$$_DQ_MP(R5) ;SET TO -1 IF GET STATUS FAILS
07E0 1746 BSBW DQ_GETSTS ;GET THE STATUS
07E0 1747 BLBC R0,20$ ;BRANCH IF GET STATUS FAILED
07E0 1748 MOVL RB MP(R4),UCB$$_DQ_MP(R5) ;SAVE MPR (STATUS WORD)
07E0 1749 BBC #RB_CS_V_CE,- ;BRANCH IF NO ERRORS (DON'T CLEAR IF
07E0 1750 UCB$$_DQ_CS(R5),30$ ;...ONLY HERE FOR DIAGNOSTIC BUFFER)
07E0 1751 BSBW DQ_RESET ;CLEAR DRIVE ERRORS IF ANY
07E0 1752 :
07E0 1753 :
07E0 1754 : SAVE UBA REGISTERS
07E0 1755 :
07E0 1756 :
07E0 1757 :
07E0 1758 30$: ASSUME UCB$$_DQ_FMPR EQ UCB$$_DQ_MP+4 ;ASSUME REGISTER AREA CONTIG
```

53 52 64 9E 07E0 1727 MOVAB RB CS(R4),R2 ;REGISTER SAVE ROUTINE  
00CC C5 9E 07E3 1728 MOVAB UCB\$\$\_DQ\_CS(R5),R3 ;GET ADDRESS OF CONTROL STATUS REGISTER  
83 82 DO 07E8 1729 MOVL (R2)+,(R3)+ ;GET ADDRESS OF REGISTER SAVE AREA  
83 82 DO 07EB 1730 MOVL (R2)+,(R3)+ ;SAVE CONTROL STATUS REGISTER  
83 82 DO 07EE 1731 MOVL (R2)+,(R3)+ ;SAVE BUFFER ADDRESS REGISTER  
83 82 DO 07F1 1732 MOVL (R2)+,(R3)+ ;SAVE BYTE COUNT REGISTER  
83 82 DO 07F4 1733 MOVL (R2)+,(R3)+ ;SAVE DISK ADDRESS REGISTER  
00C4 C5 82 F7 07F7 1734 CVTLW (R2)+,UCB\$\$\_EC1(R5) ;SAVE MPR REGISTER  
00C6 C5 82 F7 07FC 1735 CVTLW (R2)+,UCB\$\$\_EC2(R5) ;SAVE ECC POSITION REGISTER  
0801 1736 :  
0801 1737 : GET STATUS  
0801 1738 :  
0801 1739 GETUNIT ;GET UNIT NUMBER IN R2  
0809 1740 CMPZV #RB\_CS\_V\_FCODE,- ;WAS ORIGIANL FUNCTION A GET STATUS?  
080B 1741 #RB\_CS\_S\_FCODE,-  
080C 1742 UCB\$\$\_DQ\_CS(R5),-  
080F 1743 #F GETSTATUS  
0810 1744 BEQL 20\$ ;BRANCH IF SO (USE ORIGINAL STATUS)  
0812 1745 MOVZBL #-1,UCB\$\$\_DQ\_MP(R5) ;SET TO -1 IF GET STATUS FAILS  
0818 1746 BSBW DQ\_GETSTS ;GET THE STATUS  
081B 1747 BLBC R0,20\$ ;BRANCH IF GET STATUS FAILED  
081E 1748 MOVL RB MP(R4),UCB\$\$\_DQ\_MP(R5) ;SAVE MPR (STATUS WORD)  
0824 1749 BBC #RB\_CS\_V\_CE,- ;BRANCH IF NO ERRORS (DON'T CLEAR IF  
0826 1750 UCB\$\$\_DQ\_CS(R5),30\$ ;...ONLY HERE FOR DIAGNOSTIC BUFFER)  
082A 1751 BSBW DQ\_RESET ;CLEAR DRIVE ERRORS IF ANY  
082D 1752 :  
082D 1753 :  
082D 1754 : SAVE UBA REGISTERS  
082D 1755 :  
082D 1756 :  
082D 1757 :  
082D 1758 30\$: ASSUME UCB\$\$\_DQ\_FMPR EQ UCB\$\$\_DQ\_MP+4 ;ASSUME REGISTER AREA CONTIG



```
63 00000000 FFFFFFFF 8F 7D 082D 1759      MOVQ    #-1,(R3)      ;SET DEFAULT VALUE -1
                                0A 91 0838 1760      CMPB     #CDF_WRITECHECK,- ;DRIVE RELATED FUNCTION?
                                0093 C5 083A 1761      UCBSB_CEX(R5)      ;BRANCH IF SO
                                2C 1A 083D 1762
                                083F 1763
50 00D0 C5 F7 8F 78 083F 1764      ASHL     #-9,UCBSL_DQ_BA(R5),R0 ;COMPUTE MAP REGISTER NUMBER
    50 01EF 8F B1 0846 1765      CMPW     #495,R0      ;LEGAL MAP REGISTER NUMBER?
                                1E 1F 084B 1766      BLSSU    70$      ;BRANCH IF NOT
                                084D 1767
                                51 24 A5 D0 084D 1768      MOVL     UCBSL_CRB(R5),R1      ;FETCH CRB
                                52 38 B1 D0 0851 1769      MOVL     @CRBSL_INTD+VECSL_ADP(R1),R2 ;FETCH ADDRESS OF ADAPTOR CSR
83 0800 C240 D0 0855 1770      MOVL     UBISL_MAP(R2)(R0),(R3)+ ;SAVE FINAL MAP REGISTER
                                50 D7 085B 1771      DECL     R0      ;CALCULATE PREVIOUS MAP REGISTER NUMBER
                                00 EC 085D 1772      CMPV     #VECSW_MAPREG,- ;COMPARE STARTING MAP REGISTER NUMBER
                                OF 085F 1773
                                34 A1 0860 1774      CRBSL_INTD+VECSW_MAPREG(R1),- ;...FROM CRB
                                50 0862 1775      R0      ;...TO ENDING MAP REGISTER MINUS 1
                                06 14 0863 1776      BGTR     70$      ;BRANCH IF WE'RE STILL ON FIRST MAP
63 0800 C240 D0 0865 1777      MOVL     UBISL_MAP(R2)(R0),(R3) ;SAVE PREVIOUS MAP REGISTER
                                05 086B 1778 70$:      RSB      ;RETURN
                                086C 1779
```



```
086C 1781 .SBTTL UNEXPECTED INTERRUPT HANDLER
086C 1782 :++
086C 1783 :
086C 1784 : FUNCTIONAL DESCRIPTION:
086C 1785 :
086C 1786 : UNEXPECTED INTERRUPTS ARE FIELDDED. STATUS IS OBTAINED FROM
086C 1787 : THE DRIVE, VOLUME VALID IS CLEARED IF APPROPRIATE AND THE
086C 1788 : DRIVE IS RESET
086C 1789 :
086C 1790 : INPUTS:
086C 1791 :
086C 1792 : R2 - UNIT NUMBER OF DRIVE
086C 1793 : R4 - RB730 CSR ADDRESS
086C 1794 : R5 - UCB ADDRESS
086C 1795 :
086C 1796 : OUTPUTS:
086C 1797 :
086C 1798 : R0 - DESTROYED
086C 1799 : UCB$V_VALID IN UCB$W_STS
086C 1800 :
086C 1801 :--
086C 1802 :
086C 1803 DQ_UNEXINT: ;UNEXPECTED INTERRUPT
086C 1804 TSTL R5 ;IS THERE AN UCB?
086E 1805 BEQL 50$ ;BRANCH IF NOT
0870 1806
0870 1807 BSBB DQ_GETSTS ;GET DRIVE STATUS
0872 1808 BLBC R0,20$ ;BRANCH IF TIMEOUT OR OPI
0875 1809 BISW #UCB$M_ONLINE,UCB$W_STS(R5) ;SET UCB STATUS VOLUME ONLINE
0879 1810
0879 1811 BITL #RB_CS_M_TYP,RB_CS(R4) ;IS THIS AN R80
0880 1812 BNEQ 10$ ;BRANCH IF SO
0882 1813 BITL #RB_MP_M_VC,RB_MP(R4) ;VOLUME CHECK?
088A 1814 BEQL 30$ ;BRANCH IF NOT
088C 1815 BRB 20$ ;SET VOLUME INVALID
088E 1816
088E 1817 10$: BITL #RB_MP_M_PLGV,RB_MP(R4) ;PLUG VALID?
0896 1818 BNEQ 30$ ;BRANCH IF SO
0898 1819
0898 1820 20$: BBCC #UCB$V_VALID,- ;CLEAR VALID BIT
089A 1821 UCB$W_STS(R5),30$ ;...
089D 1822
089D 1823 30$: BSBW DQ_CLASSIFY ;CLASSIFY DRIVE AND INIT UCB
08A0 1824
08A0 1825 50$: BSBB DQ_RESET ;RESET THE DRIVE
08A2 1826 RSB ;RETURN TO CALLER
08A3 1827
```



```
08A3 1829 .SBTTL GET STATUS, RESET, READ HEADER
08A3 1830 :++
08A3 1831 :
08A3 1832 : DQ_READHDR - READ HEADER (EITHER DRIVE)
08A3 1833 : DQ_RESET - GET STATUS AND RESET ROUTINE
08A3 1834 : DQ_GETSTS - GET STATUS ROUTINE
08A3 1835 :
08A3 1836 : FUNCTIONAL DESCRIPTION:
08A3 1837 :
08A3 1838 : THIS ROUTINE HANDLES NON-INTERRUPT DRIVEN DEVICE OPERATIONS INCLUDING:
08A3 1839 :
08A3 1840 :     RESET DRIVE
08A3 1841 :     GET STATUS
08A3 1842 :     READ HEADER
08A3 1843 :
08A3 1844 : AFTER EXECUTING THE FUNCTION A WAIT FOR CONTROLLER READY IS DONE.
08A3 1845 : THE WAIT WILL TIMEOUT IF CONTROLLER READY DOES NOT APPEAR WITHIN
08A3 1846 : 2 SECONDS
08A3 1847 :
08A3 1848 : THIS ROUTINE SHOULD ONLY BE CALLED AT DEVICE IPL OR ABOVE
08A3 1849 :
08A3 1850 : INPUTS:
08A3 1851 :
08A3 1852 :     R2 - UNIT NUMBER IN DRIVE SELECT BITS
08A3 1853 :     R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
08A3 1854 :
08A3 1855 : OUTPUTS:
08A3 1856 :
08A3 1857 :     RB_MP(R4) - DRIVE STATUS IF DQ_GETSTS OR DQ_RESET
08A3 1858 :     R0 - LOW BIT CLEAR IF A TIMEOUT OR OPERATION INCOMPLETE
08A3 1859 :
08A3 1860 :
08A3 1861 : --
08A3 1862 :
08A3 1863 :
08A3 1864 : TO READ A HEADER, THE COMMAND IS LOADED AND THE WAIT ROUTINE
08A3 1865 : IS JUMPED TO.
08A3 1866 :
08A3 1867 : THE RB730 HOST (VAX730) MICROCODE MAINTAINS AN INTERNAL RECORD
08A3 1868 : OF THE CURRENT DISK CYLINDER FOR RB02'S. THIS REGISTER IS USED TO
08A3 1869 : COMPUTE THE RELATIVE CYLINDER ADDRESSES REQUIRED BY THE DRIVE. THE
08A3 1870 : CONTENTS OF THIS REGISTER MAY DISAGREE WITH THE ACTUAL DISK POSITION.
08A3 1871 :
08A3 1872 : THE REGISTER IS RECALIBRATED BY DOING A READ HEADER. THE MICROCODE
08A3 1873 : RELOADS THE REGISTER WITH THE CURRENT CYLINDER ADDRESS AS SPECIFIED
08A3 1874 : IN THE HEADER WORD, WHEN THE MACRO CODE READS THE MPR.
08A3 1875 :
08A3 1876 :
08A3 1877 : DQ_READHDR:                                ;DRIVE READ HEADER ENTRY
08A3 1878 :     BSBB DQ_WAIT                            ;MAKE SURE CONTROLLER FREE
08A3 1879 :     BLBS R0,10$                             ;BRANCH IF SO
08A3 1880 :     RSB                                     ;RETURN WITH R0 LBC
08A3 1881 :     BISL3 R2,-                               ;MERGE UNIT NUMBER
08A3 1882 :     #F READHEAD-                           ;...WITH FUNCTION
08A3 1883 :     !RB_CS M IE,-                           ;...AND INTERRUPT ENABLE
08A3 1884 :     RB_CS(R4)                               ;INTO CSR CLEARING CRDY
08A3 1885 :     BSBW DQ_WAIT                            ;WAIT FOR COMPLETION

3C 10 08A3 1878 BSBB DQ_WAIT
01 50 E8 08A5 1879 BLBS R0,10$
      05 08A8 1880 RSB
      52 C9 08A9 1881 BISL3 R2,-
      08AB 1882 #F READHEAD-
      08AB 1883 !RB_CS M IE,-
64 00000048 8F 08AB 1884 RB_CS(R4)
      002D 30 08B1 1885 BSBW DQ_WAIT
```



```
10 A4 10 A4 D1 08B4 1886      CMPL  RB_MP(R4),RB_MP(R4)      ;READ HEADER WORDS
18      11 08B9 1887      BRB    CHECKOPI                  ;CHECK FOR COMPLETION
08BB 1888      ;
08BB 1889      ;TO RESET THE DRIVE, A GET STATUS SUBCOMMAND IS LOADED INTO THE
08BB 1890      ;MULTIPURPOSE REGISTER WITH THE RESET BIT SET.
08BB 1891      ;
08BB 1892      ;
08BB 1893      ;
10 A4 0B D0 08BB 1894      DQ_RESET:                        ;DRIVE RESET ENTRY
08BF 1895      MOVL  #RB_MP_M_STS!-                        ;PUT GET STATUS IN MPR
08BF 1896      RB_MP_M_RST!-                                ;...AND RESET THE DRIVE
04      11 08BF 1897      BRB    RB_MP_M_MRK,RB_MP(R4)      ;...MARK SUBCOMMAND PRESENT
08C1 1898      EXGETSTS                                     ;CONTINUE IN COMMON
08C1 1899      ;
08C1 1900      ;TO GET STATUS WITHOUT RESET, A GET STATUS SUBCOMMAND IS LOADED INTO
08C1 1901      ;THE MULTIPURPOSE REGISTER. DRIVE STATUS IS NOT RESET
08C1 1902      ;
08C1 1903      ;
10 A4 03 D0 08C1 1904      DQ_GETSTS:                        ;GET STATUS ENTRY
08C5 1905      MOVL  #RB_MP_M_STS!-                        ;PUT GET STATUS IN MPR
08C5 1906      RB_MP_M_MRK,RB_MP(R4)                       ;...MARK SUBCOMMAND PRESENT
08C5 1907      ;
08C5 1908      ;NOW EXECUTE THE ACTUAL COMMAND BY MERGING THE UNIT NUMBER WITH
08C5 1909      ;THE GET STATUS COMMAND AND LOADING THE CSR. INTERRUPTS ARE NOT ENABLED
08C5 1910      ;
08C5 1911      ;
08C5 1912      EXGETSTS:                                    ;COMMAND EXECUTION
01 1A 10 08C5 1913      BSBB  DQ_WAIT                        ;MAKE SURE CONTROLLER FREE
50 E8 08C7 1914      BLBS  R0,10$                          ;BRANCH IF SO
05 08CA 1915      RSB    ;RETURN WITH R0 LBC
52 C9 08CB 1916      10$:  BISL3 R2,-                       ;MERGE UNIT NUMBER
08CD 1917      #F GETSTATUS-                                ;...WITH FUNCTION
08CD 1918      !RB_CS_M_IE,-                                ;...AND INTERRUPT ENABLE
08CD 1919      RB_CS(R4)                                     ;INTO CSR CLEARING CRDY
08D3 1920      ;
08D3 1921      CHECKOPI:                                    ;CHECK FOR OPERATION INCOMPLETE
64 00000044 8F 08D3 1922      BSBB  DQ_WAIT                  ;WAIT FOR READY
0C 10 08D5 1923      BITL  #RB_CS_M_OPI,RB_CS(R4)           ;OPERATION COMPLETE?
64 00000400 8F D3 08D5 1923      10$                        ;BRANCH IF SO
02 13 08DC 1924      BEQL  10$                               ;SET FAILURE
50 D4 08DE 1925      CLRL  R0                                ;R0 LBC IF TIMEOUT
05 08E0 1926      10$:  RSB
08E1 1927
```



```

08E1 1929 .SBTTL WAIT FOR CONTROLLER READY
08E1 1930 :++
08E1 1931 :
08E1 1932 : DQ_WAIT - WAIT FOR CONTROLLER READY ROUTINE
08E1 1933 :
08E1 1934 : FUNCTIONAL DESCRIPTION:
08E1 1935 :
08E1 1936 : THIS ROUTINE WAIT FOR CONTROLLER READY -- THE WAIT WILL TIMEOUT
08E1 1937 : IF CONTROLLER READY DOES NOT APPEAR WITHIN 2 SECONDS
08E1 1938 :
08E1 1939 : THIS ROUTINE SHOULD ONLY BE CALLED AT DEVICE IPL OR ABOVE
08E1 1940 :
08E1 1941 : INPUTS:
08E1 1942 :
08E1 1943 : R2 - UNIT NUMBER IN DRIVE SELECT BITS
08E1 1944 : R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
08E1 1945 :
08E1 1946 : OUTPUTS:
08E1 1947 :
08E1 1948 : R0 - LOW BIT CLEAR IF A TIMEOUT
08E1 1949 :
08E1 1950 :
08E1 1951 :--
08E1 1952 :
08E1 1953 :
08E1 1954 : WAIT FOR CONTROLLER READY. IF NOT PRESENT WITHIN APPROXIMATELY
08E1 1955 : 2 SECONDS, THEN R0 WILL HAVE LOW BIT CLEAR
08E1 1956 :
08E1 1957 DQ_WAIT:
08E1 1958 TIMEWAIT #200000,#RB_CS_M_CRDY,-;WAIT FOR CONTROLLER READY
08E1 1959 RB_CS(R4),L;WAIT FOR CONTROLLER READY
05 090C 1960 RSB;200000*10 MICS
090D 1961 ;RETURN TO CALLER

```



```
090D 1963      .SBTTL  UNIT INITIALIZATION ROUTINE
090D 1964
090D 1965      :++
090D 1966      :
090D 1967      DQ_UNIT_INIT - UNIT INITIALIZATION ROUTINE
090D 1968      :
090D 1969      FUNCTIONAL DESCRIPTION:
090D 1970      :
090D 1971      THIS ROUTINE READIES THE RB02/RB80 UNITS FOR I/O OPERATIONS.
090D 1972      :
090D 1973      THE OPERATING SYSTEM CALLS THIS ROUTINE:
090D 1974      - AT SYSTEM STARTUP
090D 1975      - DURING DRIVER LOADING
090D 1976      - DURING RECOVERY FROM POWER FAILURE
090D 1977      :
090D 1978      INPUTS:
090D 1979      :
090D 1980      R4      - CSR ADDRESS (CONTROLLER STATUS REGISTER)
090D 1981      R5      - UCB ADDRESS (UNIT CONTROL BLOCK)
090D 1982      :
090D 1983      OUTPUTS:
090D 1984      :
090D 1985      THE DRIVE IS RESET, UCB FIELDS ARE INITIALIZED, AND THE
090D 1986      ROUTINE WAITS FOR ONLINE UNITS TO SPIN UP.  ALL REGISTERS
090D 1987      EXCEPT R0-R3 ARE PRESERVED.  RB80'S ARE DIFFERENTIATED FROM
090D 1988      RB02 UNITS FOR UCB INITIALIZATION PURPOSES
090D 1989      :
090D 1990      A PERMANENT BUFFERED DATAPATH AND A PERMANENT SET OF MAP
090D 1991      REGISTERS ARE ALLOCATED ON THE FIRST CALL TO THIS ROUTINE.
090D 1992      ON SUCESSIVE ENTRYS, THE CALLS TO ALLOCATE RESOURCES ARE
090D 1993      IGNORED BY THE SYSTEM.
090D 1994      :--
090D 1995
090D 1996
090D 1997      DQ_UNIT_INIT:                                ;RB02/RB80 UNIT INITIALIZATION
090D 1998
090D 1999      :
090D 2000      : GET CURRENT DRIVE STATUS AND RESET DRIVE
090D 2001      :
090D 2002      MOVZWL  UCBSW_STS(R5),R3                      ;SAVE CURRENT UNIT STATUS
090D 2003      BICW    #UCBSM_ONLINE,UCBSM_VALID,-          ;ASSUME OFFLINE/INVALID
090D 2004      UCBSW_STS(R5)
090D 2005      GETUNIT
090D 2006      BSBB    DQ_WAIT                                ;LOAD UNIT NUMBER IN R2
090D 2007      BLBC    R0,50$                                ;WAIT FOR CONTROLLER
090D 2008      BSBW    DQ_RESET                                ;BRANCH IF CONTROLLER BUSY
090D 2009      BLBC    R0,50$                                ;GET STATUS AND RESET DRIVE
090D 2010
090D 2011      :
090D 2012      : WAIT FOR ONLINE UNITS TO SPIN UP
090D 2013      :
090D 2014      :
090D 2015      BBC     #UCBSV_VALID,R3,40$                  ;BYPASS SPINUP WAIT IF NOT
090D 2016      10$:    BITL    #RB_CS_M_DRDY,RB_CS(R4)        ;...VALID BEFORE POWER FAIL
090D 2017      BNEQ    30$                                     ;IS DRIVE READY?
090D 2018      JSB     G^EXESPWRTIMCHK                       ;BRANCH IF READY
090D 2019      :                                             ;IS MAX TIME EXCEEDED?
```

53 64 A5 3C 090D 2002 MOVZWL UCBSW\_STS(R5),R3 ;SAVE CURRENT UNIT STATUS  
0810 BF AA 0911 2003 BICW #UCBSM\_ONLINE,UCBSM\_VALID,- ;ASSUME OFFLINE/INVALID  
64 A5 0915 2004 UCBSW\_STS(R5)  
C0 10 0917 2005 GETUNIT ;LOAD UNIT NUMBER IN R2  
26 50 E9 091F 2006 BSBB DQ\_WAIT ;WAIT FOR CONTROLLER  
FF94 30 0921 2007 BLBC R0,50\$ ;BRANCH IF CONTROLLER BUSY  
20 50 E9 0924 2008 BSBW DQ\_RESET ;GET STATUS AND RESET DRIVE  
0927 2009 BLBC R0,50\$ ;BRANCH IF TIMEOUT OR OPI  
092A 2010  
092A 2011  
092A 2012 : WAIT FOR ONLINE UNITS TO SPIN UP  
092A 2013 :  
092A 2014 :  
16 53 0B E1 092A 2015 BBC #UCBSV\_VALID,R3,40\$ ;BYPASS SPINUP WAIT IF NOT  
64 01 D3 092E 2016 10\$: BITL #RB\_CS\_M\_DRDY,RB\_CS(R4) ;...VALID BEFORE POWER FAIL  
0B 12 092E 2017 BNEQ 30\$ ;IS DRIVE READY?  
00000000 GF 16 0931 2018 JSB G^EXESPWRTIMCHK ;BRANCH IF READY  
0933 2019 : ;IS MAX TIME EXCEEDED?



```

      F2 50  E8 0939 2020      BLBS  R0,10$      ;IF LBS - NO, STILL MORE TIME NEEDED
      06 11 093C 2021      BRB  40$      ;POWER UP TIME EXCEEDED
      64 A5 0800 8F A8 093E 2022
      64 A5 10 A8 0944 2023 30$: B1SW #UCBSM_VALID,UCBSW_STS(R5) ;SET UCB STATUS VOLUME VALID
      30 10 0948 2024 40$: B1SW #UCBSM_ONLINE,UCBSW_STS(R5) ;SET UCB STATUS VOLUME ONLINE
      094A 2025      BSBB DQ_CLASSIFY ;CLASSIFY DRIVE
      094A 2026
      094A 2027
      094A 2028
      094A 2029 : ALLOCATE A PERMANENT BUFFERED DATAPATH. (ON A VAX730, ALL DATAPATHS
      094A 2030 : ARE DIRECT. IT IS ALLOCATED HERE ONLY OUT OF CONVENTION.)
      094A 2031
      04 50  E8 0950 2032 50$: REQDPRNW ;REQUEST A PATH -- NO WAIT
      51 24 A5 D0 0953 2033      BLBS R0,55$ ;BRANCH IF SUCCESSFUL
      00 37 A1 E2 0957 2034      BUG_CHECK UBMAPEXCED,FATAL ;SERIOUS PROBLEM
      095B 2035 55$: MOVCL UCB$L_CRB(R5),R1 ;FETCH CRB ADDRESS
      095D 2036      BBSS #VEC$D_PATHLOCK,- ;LOCK THE DATA PATH
      0960 2037      CRB$L_INTD+VEC$B_DATAPATH(R1),65$ ;...IN THE CRB
      0960 2038
      0960 2039 :
      0960 2040 : ACCOCATE ENOUGH PERMANENT MAP REGISTERS TO HOLD THE LARGEST POSSIBLE
      0960 2041 : DATA TRANSFER. SINCE NEITHER THE RB02 OR RB80 SUPPORT SPIRALLING THE
      0960 2042 : LARGEST TRANSFER IS A SINGLE TRACK
      0960 2043 :
      0960 2044 : 32 (RB80 BLOCKS PER TRACK) +
      0960 2045 : 1 (IN CASE TRANSFER CROSSES PAGE BOUND) +
      0960 2046 : 1 (FOR INVALID SENTINAL PAGE)
      0960 2047 : -- =
      0960 2048 : 34 PERMANENTLY ALLOCATED MAP REGISTERS
      0960 2049 :
      53 22  D0 0960 2050 65$: MOVCL #34,R3 ;34 MAP REGISTERS NEEDED
      00000000 GF 16 0963 2051      JSB G^IOCSALOUBAMAPN ;REQUEST THEM
      0969 2052
      04 50  E8 0969 2053      BLBS R0,67$ ;BRANCH IF SUCCESSFUL
      096C 2054      BUG_CHECK UBMAPEXCED,FATAL ;SERIOUS PROBLEM
      0970 2055
      51 24 A5 D0 0970 2056 67$: MOVCL UCB$L_CRB(R5),R1 ;FETCH CRB ADDRESS
      00 34 A1 E2 0974 2057      BBSS #VEC$D_MAPLOCK,- ;LOCK THE MAPS
      0976 2058      CRB$L_INTD+VEC$W_MAPREG(R1),75$ ;...IN THE CRB
      05 0979 2059 75$: RSB ;RETURN
      097A 2060
```



```
097A 2062 .SBTTL DRIVE CLASSIFICATION ROUTINE
097A 2063 :++
097A 2064 :
097A 2065 DQ_CLASSIFY - DRIVE CLASSIFICATION ROUTINE
097A 2066 :
097A 2067 FUNCTIONAL DESCRIPTION:
097A 2068 :
097A 2069 THIS ROUTINE IS CALLED TO CLASSIFY THE DRIVE TYPE AND INITIALIZE
097A 2070 THE UCB FEILDS. IT IS CALLED AT DRIVE INIT TIME, AND FOLLOWING AN
097A 2071 UNEXPECT INTERRUPT.
097A 2072 :
097A 2073 INPUTS:
097A 2074 :
097A 2075 R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
097A 2076 R5 - ADDRESS OF UNIT CONTROL BLOCK (UCB)
097A 2077 :
097A 2078 OUTPUTS:
097A 2079 :
097A 2080 R0-R2 - DESTROYED
097A 2081 THE UCB FEILDS ARE INITIALIZED
097A 2082 :
097A 2083 :--
097A 2084 :
097A 2085 DQ_CLASSIFY: ;DRIVE CLASSIFICATION ROUTINE
097A 2086 :
097A 2087 ASSUME THAT SECTORS, TRACKS, AND CYLINDERS FEILDS ARE CONTAINED IN
097A 2088 UCB$$_DEVDEPEND
097A 2089 :
097A 2090 ASSUME UCB$$_SECTORS EQ UCB$$_DEVDEPEND
097A 2091 ASSUME UCB$$_TRACKS EQ UCB$$_DEVDEPEND+1
097A 2092 ASSUME UCB$$_CYLINDERS EQ UCB$$_DEVDEPEND+2
097A 2093 :
097A 2094 :
097A 2095 ASSUME ITS AN RB02 AND INITIALIZE ACCORDINGLY
097A 2096 :
097A 2097 MOVB #DTS_RB02,UCB$$_DEVTYPE(R5) ;SET RB02 DEVICE TYPE AND
097E 2098 MOVL #<407<208>+<512016>>,- ;LOAD SECTORS+TRACKS+CYLINDERS
0984 2099 UCB$$_DEVDEPEND(R5) ;... INTO UCB
0986 2100 MOVZWL #<20*2*512>,UCB$$_MAXBLOCK(R5) ;((512 BYTE) BLOCKS PER SPINDLE
098D 2101 MOVL #*X24642002,UCB$$_MEDIA_ID(R5) ;SET MEDIA IDENT "DQ RB02"
0996 2102 BISW2 #UCB$$_NOCNVRT,UCB$$_DEVSTS(R5) ;DISABLE LOG TO PHYS CONV.
099A 2103 GETUNIT ;PUT UNIT NUMBER IN R2
09A2 2104 BSBW DQ_READHDR ;READ HEADER TO SYNCRONIZE UCODE
09A5 2105 :
09A5 2106 BITL #RB_CS_M_TYP,RB_CS(R4) ;TEST DRIVE TYPE
09AC 2107 BEQL 30$ ;BRANCH IF AN RB02
09AE 2108 :
09AE 2109 MOVB #DTS_RB80,UCB$$_DEVTYPE(R5) ;SET RB80 DEVICE TYPE AND
09B2 2110 MOVL #<317<1408>+<559016>>,- ;LOAD SECTORS+TRACKS+CYLINDERS
09B8 2111 UCB$$_DEVDEPEND(R5) ;... INTO UCB
09BA 2112 MOVL #<31*4*559>,UCB$$_MAXBLOCK(R5) ;((512 BYTE) BLOCKS PER SPINDLE
09C3 2113 MOVL #*X24642050,UCB$$_MEDIA_ID(R5) ;SET MEDIA IDENT "DQ RB80"
09CC 2114 BICW2 #UCB$$_NOCNVRT,UCB$$_DEVSTS(R5) ;ENABLE LOG TO PHYS CONV.
09D0 2115 :
09D0 2116 30$: RSB
09D1 2117 :
```

41 A5 12 90	02000228 8F D0	44 A5 3C	0080 C5 5000 8F D0	24642002 8F D0	68 A5 04 A8	FEFE 30	64 04000000 8F D3	22 13	41 A5 13 90	022F0E1F 8F D0	44 A5 09B8	0080 C5 0003B3AE 8F D0	008C C5 24642050 8F D0	68 A5 04 AA	09D0 2115	05 09D0 2116	09D1 2117
-------------	----------------	----------	--------------------	----------------	-------------	---------	-------------------	-------	-------------	----------------	------------	------------------------	------------------------	-------------	-----------	--------------	-----------



```
09D1 2119 .SBTTL CONTROLLER INITIALIZATION ROUTINE
09D1 2120 :++
09D1 2121 :
09D1 2122 FUNCTIONAL DESCRIPTION:
09D1 2123 :
09D1 2124 THE CORRECT RB730 CSR ADDRESS IS COMPUTED.
09D1 2125 :
09D1 2126 AUTOCONFIGURE UTILIZES A TEMPORARY UNIBUS CSR ADDRESS FOR
09D1 2127 CONFIGURING THIS CONTROLLER. THE TRUE CSR ADDRESS IS LOCATED
09D1 2128 EXACTLY ONE PAGE ABOVE THE ADAPTOR CSR (THE DEVICE REGISTERS
09D1 2129 ARE ACTUALLY IN THE ADAPTOR CONTROL REGISTER REGION)
09D1 2130 :
09D1 2131 THE OPERATING SYSTEM CALLS THIS ROUTINE:
09D1 2132 - AT SYSTEM STARTUP
09D1 2133 - DURING DRIVER LOADING
09D1 2134 - DURING RECOVERY FROM POWER FAILURE
09D1 2135 :
09D1 2136 INPUTS:
09D1 2137 :
09D1 2138 R4 - CSR ADDRESS (DEVICE CONTROL STATUS REGISTER)
09D1 2139 R5 - IDB ADDRESS (INTERRUPT DATA BLOCK)
09D1 2140 R6 - DDB ADDRESS (DEVICE DATA BLOCK)
09D1 2141 R8 - CRB ADDRESS (CHANNEL REQUEST BLOCK)
09D1 2142 ALL INTERRUPTS ARE LOCKED OUT
09D1 2143 :
09D1 2144 OUTPUTS:
09D1 2145 :
09D1 2146 IDB$$_CSR - CORRECT RB730 CSR ADDRESS
09D1 2147 :
09D1 2148 :--
09D1 2149 :
09D1 2150 DQ_RB730 INIT: ;CONTROLLER INITIALIZATION
09D1 2151 ASSUME ADP$$_CSR EQ 0
09D1 2152 MOVL @IDB$$_ADP(R5),R0 ;FETCH ADAPTOR CSR ADDRESS
09D5 2153 MOVAL ^X200(R0),IDB$$_CSR(R5) ;STORE CSR IN IDB
09DA 2154 RSB
09DB 2155
```

50 14 B5 D0  
65 0200 C0 DE  
05



```

09DB 2157 .SBTTL UNIT DELIVERY ROUTINE
09DB 2158 :++
09DB 2159 :
09DB 2160 DQ_DELIVER - UNIT DELIVERY ROUTINE
09DB 2161 :
09DB 2162 FUNCTIONAL DESCRIPTION:
09DB 2163 :
09DB 2164 THIS ROUTINE IS CALLED BY AUTOCONFIGURE TO TEST FOR A UNITS
09DB 2165 PRESCENCE OR ABSCEANCE ON THE CONTROLLER
09DB 2166 :
09DB 2167 INPUTS:
09DB 2168 :
09DB 2169 R0-R3 - SCRATCH
09DB 2170 R4 - ADDRESS OF ADAPTOR CONFIGURATION REGISTER
09DB 2171 R5 - UNIT NUMBER TO BE CONFIGURED
09DB 2172 R6 - ADDRESS OF CONFIGURATION CONTROL REGISTER
09DB 2173 R7 - ADDRESS OF CONFIGURATION CONTROL BLOCK (ACF BLOCK)
09DB 2174 R8 - ADDRESS OF ADAPTOR CONTROL BLOCK
09DB 2175 :
09DB 2176 ACF$B_CUNIT(R7) - UNIT NUMBER TO BE TESTED
09DB 2177 :
09DB 2178 OUTPUTS:
09DB 2179 :
09DB 2180 R0 - LBS IF UNIT FOUND, LBC IF NO SUCH UNIT
09DB 2181 :
09DB 2182 :--
09DB 2183 :
09DB 2184 DQ_DELIVER: ;UNIT DELIVERY ROUTINE
54 0200 C6 9E 09DB 2185 PUSHL R4 ;SAVE R4
52 02 08 52 D4 09DD 2186 MOVAB ^X200(R6),R4 ;COMPUTE ADDRESS OF CSR
FEF5 30 09E2 2187 CLRL R2 ;PREPARE FOR UNIT NUMBER
03 50 F0 09E4 2188 INSV R5,#8,#2,R2 ;LOAD DRIVE SELECT BITS
FECF 30 09E9 2189 BSBW DQ_WAIT ;WAIT FOR CONTROLLER READY
09EC 2190 BLBC R0,50$ ;BRANCH IF CONTROLLER BUSY
09EF 2191 BSBW DQ_GETSTS ;ATTEMPT GET STATUS
09F2 2192 ;R0=1 IF OK, 0 IF NO UNIT
09F2 2193 50$: BICL #RB_CS_M_IE,RB_CS(R4) ;DISABLE INTERRUPTS
09F9 2194 MOVL (SP)+,R4 ;RESTORE R4
09FC 2195 RSB ;RETURN STATUS TO CALLER
09FD 2196
09FD 2197

```



```
09FD 2199 .SBTTL REGISTER DUMP ROUTINE
09FD 2200 :++
09FD 2201 :
09FD 2202 DQ_REGDUMP - REGISTER DUMP ROUTINE
09FD 2203 :
09FD 2204 FUNCTIONAL DESCRIPTION:
09FD 2205 :
09FD 2206 THIS ROUTINE IS CALLED TO SAVE THE DEVICE REGISTERS AND UBA RESOURCE
09FD 2207 REGISTERS IN A SPECIFIED BUFFER. IT IS CALLED FROM THE DEVICE ERROR
09FD 2208 LOGGING ROUTINE AND FROM THE DIAGNOSTIC BUFFER FILL ROUTINE.
09FD 2209 :
09FD 2210 INPUTS:
09FD 2211 :
09FD 2212 R0 - ADDRESS OF REGISTER SAVE BUFFER
09FD 2213 R4 - ADDRESS OF DEVICE CONTROL STATUS REGISTER (CSR)
09FD 2214 R5 - ADDRESS OF UNIT CONTROL BLOCK (UCB)
09FD 2215 :
09FD 2216 OUTPUTS:
09FD 2217 :
09FD 2218 THE DEVICE AND UBA REGISTERS ARE SAVED IN THE SPECIFIED BUFFER.
09FD 2219 R0 CONTAINS THE ADDRESS OF THE NEXT EMPTY LONGWORD IN THE BUFFER.
09FD 2220 ALL REGISTERS EXCEPT R1 AND R2 ARE PRESERVED.
09FD 2221 :
09FD 2222 :--
09FD 2223 :
09FD 2224 DQ_REGDUMP:
09FD 2225 MOVL #<RB_NUM_REGS+6>,(R0)+ ;REGISTER DUMP ROUTINE
51 80 0D D0 09FD 2226 MOVAL UCB$C_DQ_CS(R5),R1 ;INSERT NUMBER OF REGISTERS
00CC C5 DE 0A00 2227 MOVZBL #<RB_NUM_REGS-2>,R2 ;GET ADDRESS OF SAVED DEVICE REGISTERS
52 05 9A 0A05 2228 10$: MOVL (R1)+,(R0)+ ;GET NUMBER OF DEVICE REGISTERS TO MOVE
80 81 D0 0A08 2229 SOBGTR R2,10$ ;DUMP REGISTER IN BUFFER
FA 52 F5 0A0B 2230 ;IF GTR - STILL MORE TO MOVE
0A0E 2231 :
80 00C4 C5 3C 0A0E 2232 MOVZWL UCB$W_EC1(R5),(R0)+ ;ECC POSITION REGISTER
80 00C6 C5 3C 0A13 2233 MOVZWL UCB$W_EC2(R5),(R0)+ ;ECC PATTERN REGISTER
0A18 2234 :
52 24 A5 D0 0A18 2235 MOVL UCB$L_CRB(R5),R2 ;FETCH CRB ADDRESS
80 37 A2 9A 0A1C 2236 MOVZBL CRB$L_INTD+VEC$B_DATAPATH(R2),(R0)+ ;DUMP DATAPATH NUMBER
80 80 D4 0A20 2237 CLRL (R0)+ ;DUMP DATAPATH REGISTER (ALWAYS 0)
80 81 D0 0A22 2238 MOVL (R1)+,(R0)+ ;DUMP FINAL MAP REGISTER
80 81 D0 0A25 2239 MOVL (R1)+,(R0)+ ;DUMP PREVIOUS MAP REGISTER
0A28 2240 :
80 34 A2 D0 0A28 2241 ASSUME VEC$B_NUMREG EQ VEC$W_MAPREG+2 ;ASSUME START AND NUMBER CONTIG
0A28 2242 MOVL CRB$L_INTD+VEC$W_MAPREG(R2),(R0)+ ;DUMP MAP REGISTERS
0A2C 2243 :
80 00F6 C5 D0 0A2C 2244 MOVL UCB$L_DQ_PREVDA(R5),(R0)+ ;DUMP PREVIOUS DISK ADDRESS
0A31 2245 :
05 0A31 2246 RSB ;RETURN
0A32 2247 :
0A32 2248 DQ_END: ;ADDRESS OF LAST LOCATION IN DRIVER
.END
```



DQDRIVER  
Symbol table

D 16  
- VAX/VMS RB730:RB02/RB80 DISK DRIVER

15-SEP-1984 23:49:22 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:46 [DRIVER.SRC]DQDRIVER.MAR;1

Page 51  
(1)

```

$$$ = 00000020 R 02
$$OP = 00000002
ACPSACCESS ***** X 03
ACPSDEACCESS ***** X 03
ACPSMODIFY ***** X 03
ACPSMOUNT ***** X 03
ACPSREADBLK ***** X 03
ACPSWRITEBLK ***** X 03
ADPSL_CSR = 00000000
APPLY_ECC = 000002A1 R 03
ATS_UBA = 00000001
AVAILABLE = 000001E2 R 03
BRW_RETREG = 000005B7 R 03
BUGS_UBMAPEXCED ***** X 03
CDF_AVAILABLE = 00000011
CDF_DRVCLR = 00000004
CDF_NOP = 00000000
CDF_OFFSET = 00000006
CDF_PACKACK = 00000008
CDF_READDATA = 0000000C
CDF_READHEAD = 0000000E
CDF_READTRACKD = 00000010
CDF_RECAL = 00000003
CDF_RELEASE = 00000005
CDF_RETCENTER = 00000007
CDF_SEEK = 00000002
CDF_STARTSPNDL = 00000009
CDF_UNLOAD = 00000001
CDF_WRITECHECK = 0000000A
CDF_WRITEDATA = 0000000B
CDF_WRITEHEAD = 0000000D
CDF_WRITETRACKD = 0000000F
CHECKECC = 00000235 R 03
CHECKOPI = 000008D3 R 03
CRBSL_INTD = 00000024
DCS_DISK = 00000001
DDBSL_PACK = 00000001
DDBSL_ACPD = 00000010
DDBSL_DDT = 0000000C
DEVSM_AVL = 00040000
DEVSM_DIR = 00000008
DEVSM_ELG = 00400000
DEVSM_FOD = 00004000
DEVSM_IDV = 04000000
DEVSM_NHM = 00000200
DEVSM_ODV = 08000000
DEVSM_RND = 10000000
DEVSM_SHR = 00010000
DPTSC_LENGTH = 00000038
DPTSC_VERSION = 00000004
DPT$INITAB = 00000038 R 02
DPT$M_SVP = 00000002
DPT$REINITAB = 0000006D R 02
DPT$TAB = 00000000 R 02
DQ$DDT = 00000000 RG 03
DQ_CLASSIFY = 0000097A R 03
DQ_DELIVER = 000009DB R 03

```

```

DQ_END 00000A32 R 03
DQ_FUNCABLE 00000080 R 03
DQ_GETSTS 000008C1 R 03
DQ_INT 0000074D R 03
DQ_RB730_INIT 000009D1 R 03
DQ_READHDR 000008A3 R 03
DQ_REGDUMP 000009FD R 03
DQ_REGSAVE 000007E0 R 03
DQ_REI 00000747 R 03
DQ_RESET 000008BB R 03
DQ_STARTIO 00000114 R 03
DQ_UNEXINT 0000086C R 03
DQ_UNIT_INIT 0000090D R 03
DQ_WAIT 000008E1 R 03
DRCLR 00000398 R 03
DRVCLR 000001F0 R 03
DTS_RB02 = 00000012
DTS_RB80 = 00000013
DYN$C_CRB = 00000005
DYN$C_DDB = 00000006
DYN$C_DPT = 0000001E
DYN$C_UCB = 00000010
EMBSL_DV_REGSAV = 0000004E
ERL$DEVICERR ***** X 03
ERL$DEVICTMO ***** X 03
EXESGL_TENUSEC ***** X 03
EXESGL_UBDELAY ***** X 03
EXESIOFORK ***** X 03
EXESLCLDSKVALID ***** X 03
EXESONEPARM ***** X 03
EXESPWRTIMCHK ***** X 03
EXESSENSEMODE ***** X 03
EXESSETCHAR ***** X 03
EXESZEROPARM ***** X 03
EXGETSTS 000008C5 R 03
EX_IMED 000003B0 R 03
FATAL 00000744 R 03
FATALERR 000002C2 R 03
FDISPATCH 0000018A R 03
FEXL 0000035A R 03
FTAB 00000038 R 03
FUNCTAB_LEN = 00000094
FUNCXT 00000321 R 03
F_AVAILABLE = 00000004
F_DRVCLR = 00000004
F_GETSTATUS = 00000004
F_NOP = 00000000
F_OFFSET = 00000000
F_PACKACK = 00000004
F_READDATA = 0000000C
F_READHEAD = 00000008
F_READTRACKD = 00000000
F_RECAL = 00000006
F_RELEASE = 00000000
F_RETCENTER = 00000000
F_SEEK = 00000006
F_STARTSPNDL = 00000000

```



DQDRIVER  
Symbol table

E 16  
- VAX/VMS RB730:RB02/RB80 DISK DRIVER

15-SEP-1984 23:49:22 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:46 [DRIVER.SRC]DQDRIVER.MAR;1

Page 52  
(1)

F_UNLOAD	= 00000004		
F_WRITECHECK	= 00000002		
F_WRITEDATA	= 0000000A		
F_WRITEHEAD	= 00000000		
F_WRIETRACKD	= 00000000		
IDBSL_ADP	= 00000014		
IDBSL_CSR	= 00000000		
IDBSL_OWNER	= 00000004		
IDBSL_UCBLST	= 00000018		
IMMED	= 000003AD	R	03
IOSM_DATACHECK	= 00004000		
IOSV_DATACHECK	= 0000000E		
IOSV_INHRETRY	= 0000000F		
IOSV_SKPSECINH	= 00000009		
IOS_ACCESS	= 00000032		
IOS_ACPCONTROL	= 00000038		
IOS_AVAILABLE	= 00000011		
IOS_CREATE	= 00000033		
IOS_DEACCESS	= 00000034		
IOS_DELETE	= 00000035		
IOS_DRVCLR	= 00000004		
IOS_MODIFY	= 00000036		
IOS_MOUNT	= 00000039		
IOS_NOP	= 00000000		
IOS_PACKACK	= 00000008		
IOS_READHEAD	= 0000000E		
IOS_READBLK	= 00000021		
IOS_READPBLK	= 0000000C		
IOS_READVBLK	= 00000031		
IOS_RECAL	= 00000003		
IOS_SEEK	= 00000002		
IOS_SENSECHAR	= 0000001B		
IOS_SENSEMODE	= 00000027		
IOS_SETCHAR	= 0000001A		
IOS_SETMODE	= 00000023		
IOS_UNLOAD	= 00000001		
IOS_VIRTUAL	= 0000003F		
IOS_WRITECHECK	= 0000000A		
IOS_WRITEHEAD	= 0000000D		
IOS_WRI TELBLK	= 00000020		
IOS_WRITEPBLK	= 0000000B		
IOS_WRITEVBLK	= 00000030		
IOCSALOU BMAPN	*****	X	03
IOCSAPPLYECC	*****	X	03
IOCS\$DIAGBUF ILL	*****	X	03
IOCSLOADUBAMAPA	*****	X	03
IOCSMNTVER	*****	X	03
IOCSMOVTOUSER	*****	X	03
IOCSRELCHAN	*****	X	03
IOCSREQCOM	*****	X	03
IOCSREQDATAPNW	*****	X	03
IOCSREQPCHANL	*****	X	03
IOCSRETURN	*****	X	03
IOCSWFIKPC	*****	X	03
IPL\$ POWER	= 0000001F		
IRPSL_MEDIA	= 00000038		
IRPSL_SVAPTE	= 0000002C		

IRPSS_FCODE	= 00000006		
IRPSV_DIAGBUF	= 00000007		
IRPSV_FCODE	= 00000000		
IRPSV_PHYSIO	= 00000008		
IRPSW_BCNT	= 00000032		
IRPSW_FUNC	= 00000020		
IRPSW_STS	= 0000002A		
MASKH	= 00000008		
MASKL	= 04000000		
NOMAPS	= 00000517	R	03
NOP	= 000001E7	R	03
NORMAL	= 00000225	R	03
OFFSET	= 000001E7	R	03
PACKACK	= 000001D4	R	03
POSIT	= 00000421	R	03
PR\$ IPL	= 00000012		
PREPROCESS	= 0000011B	R	03
PWRFAIL	= 000006A0	R	03
RB_BA	= 00000004		
RB_BC	= 00000008		
RB_CMD	= 0000001C		
RB_CS	= 00000000		
RB_CS_M_ASSI	= 08000000		
RB_CS_M_ATN	= 000F0000		
RB_CS_M_CE	= 00008000		
RB_CS_M_CRDY	= 00000080		
RB_CS_M_DCK	= 00000800		
RB_CS_M_DE	= 00004000		
RB_CS_M_DLT	= 00001000		
RB_CS_M_DRDY	= 00000001		
RB_CS_M_FMT	= 20000000		
RB_CS_M_IE	= 00000040		
RB_CS_M_IR	= 01000000		
RB_CS_M_NXM	= 00002000		
RB_CS_M_OPI	= 00000400		
RB_CS_M_SSE	= 00800000		
RB_CS_M_SSEI	= 00400000		
RB_CS_M_TYP	= 04000000		
RB_CS_S_ATN	= 00000004		
RB_CS_S_ECS	= 00000002		
RB_CS_S_FCODE	= 00000003		
RB_CS_V_ATN	= 00000010		
RB_CS_V_CE	= 0000000F		
RB_CS_V_DCK	= 0000000B		
RB_CS_V_DE	= 0000000E		
RB_CS_V_DS	= 00000008		
RB_CS_V_ECS	= 00000014		
RB_CS_V_FCODE	= 00000001		
RB_CS_V_NXM	= 0000000D		
RB_CS_V_OPI	= 0000000A		
RB_CS_V_SSE	= 00000017		
RB_DA	= 0000000C		
RB_EC1	= 00000014		
RB_EC2	= 00000018		
RB_MP	= 00000010		
RB_MP_C_SLM	= 00000005		
RB_MP_M_BH	= 00000008		



DQDRIVER  
Symbol table

F 16  
- VAX/VMS RB730:RB02/RB80 DISK DRIVER

15-SEP-1984 23:49:22 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:46 [DRIVER.SRC]DQDRIVER.MAR;1

Page 53  
(1)

RB_MP_M_DSE	= 00000100		
RB_MP_M_HCE	= 00004000		
RB_MP_M_HO	= 00000010		
RB_MP_M_MRK	= 00000001		
RB_MP_M_PLGV	= 00000200		
RB_MP_M_RST	= 00000008		
RB_MP_M_SPD	= 00000800		
RB_MP_M_STS	= 00000002		
RB_MP_M_VC	= 00000200		
RB_MP_M_WDE	= 00008000		
RB_MP_M_WGE	= 00000400		
RB_MP_V_PLGV	= 00000009		
RB_MP_V_VC	= 00000009		
RB_MP_V_WGE	= 0000000A		
RB_MP_V_WL	= 0000000D		
RB_MP_V_WTP	= 0000000D		
RB_NUM_REGS	= 00000007		
READDATA	000001FD	R	03
READHEAD	000001F6	R	03
READTRACKD	000001E7	R	03
RECAL	000001F0	R	03
RECALB	000003E6	R	03
RELEASE	000001E7	R	03
RESETDRIVE	000002B8	R	03
RETCENTER	000001E7	R	03
RETHDR	000005BA	R	03
RETREG	000006B0	R	03
RETRY	00000728	R	03
RETRYERR	000002AA	R	03
SEEK	000001F0	R	03
SEEK1	00000444	R	03
SIZ...	= 00000020		
SPECOND	00000672	R	03
SSS_CTRLERR	= 00000054		
SSS_DATACHECK	= 0000005C		
SSS_DRVERR	= 0000008C		
SSS_MEDOFL	= 000001A4		
SSS_NORMAL	= 00000001		
SSS_PARITY	= 000001F4		
SSS_TIMEOUT	= 0000022C		
SSS_VOLINV	= 00000254		
SSS_WASECC	= 00000639		
SSS_WRITLCK	= 0000025C		
STARTSPNDL	000001E7	R	03
SUCCESS	0000073C	R	03
TRANSFER	00000221	R	03
UBISL_MAP	= 00000800		
UCBSB_CEX	= 00000093		
UCBSB_DEVCLASS	= 00000040		
UCBSB_DEVTYPE	= 00000041		
UCBSB_DIPL	= 0000005E		
UCBSB_DQ_FLAGS	= 000000C9		
UCBSB_ERTCNT	= 00000080		
UCBSB_ERTMAX	= 00000081		
UCBSB_FEX	= 00000092		
UCBSB_FIPL	= 0000000B		
UCBSB_SECTORS	= 00000044		

UCBSB_TRACKS	= 00000045
UCBSK_DQ_LEN	= 000000FA
UCBSK_LCC_DISK_LENGTH	= 000000CC
UCBSL_CRB	= 00000024
UCBSL_DEVCHAR	= 00000038
UCBSL_DEVCHAR2	= 0000003C
UCBSL_DEVDEPEND	= 00000044
UCBSL_DPC	= 0000009C
UCBSL_DQ_BA	000000D0
UCBSL_DQ_BC	000000D4
UCBSL_DQ_CS	000000CC
UCBSL_DQ_CURDA	000000F2
UCBSL_DQ_DA	000000D8
UCBSL_DQ_DPR	000000E8
UCBSL_DQ_FMPR	000000E0
UCBSL_DQ_MP	000000DC
UCBSL_DQ_PMPR	000000E4
UCBSL_DQ_PREVDA	000000F6
UCBSL_FPC	= 0000000C
UCBSL_FR3	= 00000010
UCBSL_IRP	= 00000058
UCBSL_MAXBLOCK	= 000000B0
UCBSL_MEDIA	= 000000BC
UCBSL_MEDIA_ID	= 0000008C
UCBSL_SVAPTE	= 00000078
UCBSM_DIAGBUF	= 00000002
UCBSM_DQ_DIP	= 00000002
UCBSM_DQ_ECC_DEFER	= 00000004
UCBSM_DQ_SIP	= 00000001
UCBSM_ECC	= 00000001
UCBSM_NOCNVRT	= 00000004
UCBSM_ONLINE	= 00000010
UCBSM_POWER	= 00000020
UCBSM_TIMEOUT	= 00000040
UCBSM_VALID	= 00000800
UCBSV_DIAGBUF	= 00000001
UCBSV_DQ_DIP	= 00000001
UCBSV_DQ_ECC_DEFER	= 00000002
UCBSV_DQ_SIP	= 00000000
UCBSV_ECC	= 00000000
UCBSV_INT	= 00000001
UCBSV_POWER	= 00000005
UCBSV_VALID	= 0000000B
UCBSW_BCNT	= 0000007E
UCBSW_BCR	= 000000C0
UCBSW_BOFF	= 0000007C
UCBSW_CYLINDERS	= 00000046
UCBSW_DA	= 000000BC
UCBSW_DC	= 000000BE
UCBSW_DEVBUFSIZ	= 00000042
UCBSW_DEVSTS	= 00000068
UCBSW_DQ_HDR1	000000EC
UCBSW_DQ_HDR2	000000EE
UCBSW_DQ_HDR3	000000F0
UCBSW_ECT	= 000000C4
UCBSW_EC2	= 000000C6
UCBSW_FUNC	= 0000009A



DQDRIVER  
Symbol table

G 16  
- VAX/VMS RB730:RB02/RB80 DISK DRIVER

15-SEP-1984 23:49:22 VAX/VMS Macro V04-00  
5-SEP-1984 00:12:46 [DRIVER.SRC]DQDRIVER.MAR;1

Page 54  
(1)

UCBSW_OFFSET	=	000000C8		
UCBSW_STS	=	00000064		
UCBSW_UNIT	=	00000054		
UNLOAD		000001E2	R	03
UPDATE		00000605	R	03
VECSB_DATAPATH	=	00000013		
VECSB_NUMREG	=	00000012		
VECSL_ADP	=	00000014		
VECSL_IDB	=	00000008		
VECSL_INITIAL	=	0000000C		
VECSL_UNITINIT	=	00000018		
VECSS_MAPREG	=	0000000F		
VECSV_MAPLOCK	=	0000000F		
VECSV_MAPREG	=	00000000		
VECSV_PATHLOCK	=	00000007		
VECSW_MAPREG	=	00000010		
WRITECHECK		000001F6	R	03
WRITECHK		000005EE	R	03
WRITEDATA		000001FD	R	03
WRITEHEAD		000001F0	R	03
WRITETRACKD		000001E7	R	03
XFER		000004DA	R	03
_TMP\$VAL	=	00000044		

+-----+  
! Psect synopsis !  
+-----+

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 ( 0.)	00 ( 0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS	000000FA ( 250.)	01 ( 1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
\$\$\$105_PROLOGUE	00000082 ( 130.)	02 ( 2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
\$\$\$115_DRIVER	00000A32 ( 2610.)	03 ( 3.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	LONG				

+-----+  
! Performance indicators !  
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.05	00:00:01.69
Command processing	115	00:00:00.36	00:00:05.99
Pass 1	634	00:00:20.22	00:02:22.64
Symbol table sort	0	00:00:02.57	00:00:14.43
Pass 2	387	00:00:04.98	00:00:31.10
Symbol table output	22	00:00:00.21	00:00:01.35
Psect synopsis output	0	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1191	00:00:28.41	00:03:17.22

The working set limit was 2550 pages.  
166368 bytes (325 pages) of virtual memory were used to buffer the intermediate code.  
There were 130 pages of symbol table space allocated to hold 2370 non-local and 86 local symbols.  
2248 source lines were read in Pass 1, producing 23 object records in Pass 2.  
58 pages of virtual memory were used to define 55 macros.



-----  
! Macro library statistics !  
-----

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	33
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	44

2514 GETS were required to define 44 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:DQDRIVER/OBJ=OBJ\$:DQDRIVER MSRC\$:DQDRIVER/UPDATE=(ENH\$:DQDRIVER)+EXECMLS/LIB



0109 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

